A Parallel Finite Element Solver for Large-Scale 3-D Stokes Problem and its Application to Earth's Mantle Convection Problem

Atsushi Suzuki

A dissertation submitted in partial fulfillment of the requirements
for the degree of Doctor of Informatics

Kyoto University

November 2003

Acknowledgments

Fist of all, I wish to express my sincere gratitude to Professor Tatsuo Nogi of Kyoto University. He encouraged me constantly and gave me valuable comments and suggestions throughout the whole period of the preparation for the thesis.

I also wish to express my thanks to members of the refereeing committee of the thesis, Professor Yuusuke Iso, Professor Mitsuaki Funakoshi, Professor Yutaka Yamamoto, and Professor Masayoshi Kubo.

I would like to thank Professor Masahisa Tabata of Kyushu University, who also instructed me and gave me fruitful suggestions in a joint work on development of finite element scheme and code for Earth's mantle convection problem.

I would also like to thank Professor Masato Kimura of Kyushu University, who contributed to improve the thesis through helpful discussions.

I wish to thank Professor Satoru Honda of the University of Tokyo, who introduced me Earth's mantle convection problem.

Contents

List of Figures

List of Tables

General Introduction

The aim of this thesis is to develop a parallel finite element solver for a three-dimensional (3-D) Stokes problem, by means of which numerical simulations of the Earth's mantle convection problem can be realized. A mathematical model of the Earth's mantle convection is described by the Rayleigh–Bénard equations with an infinite Prandtl number. These equations consist of the Stokes equations and the convection-diffusion equation with their nonlinear coupling. From the geometrical characteristics of the problems of the Earth's mantle convection, we consider a specific Stokes problem, that is, the Stokes equations in a spherical shell domain with slip boundary conditions imposed.

Finite element methods (FEM) [15, 5, 52] are widely used to solve elasticity problems and flow problems. FEM has the following advantages: (i) It is applicable to various domain shapes. (ii) It can treat many types of boundary conditions simply. (iii) It is obtained by discretization of a weak form, and the existence and uniqueness of a finite element solution is guaranteed in a mathematical framework of variational problems. Furthermore, it is possible to obtain an error estimate of the finite element solution. However, direct use of some techniques that are used in conventional finite element codes causes some difficulties in construction of the finite element solver for the Stokes problem.

For real 3-D computations, even with current computer resources, it is necessary to develop an efficient algorithm for fast computation with small memory requirements. It is also desirable that the algorithm be consistent with the framework of the finite element method and be easily implemented. The nature of easy implementation assists parallelization of the algorithm.

We describe five problems that may be caused by the techniques of conventional finite element codes and our remedies for each problem in the following.

(i) The Stokes equations are treated with a weak form of the saddle-point type. In the mixed finite element approximation, where the velocity and the pressure are discretized simultaneously, it is necessary to use a pair of elements satisfying the so-called inf-sup condition that ensures the solvability of the discretized equations [23, 13, 10]. For this purpose, P2/P1 or iso-P2 P1/P1 elements [23, 13] are commonly used. However, these elements require considerable memory to store the stiffness matrix.

We use the same interpolation for both the velocity and pressure by P1 elements, where unknowns are approximated by piecewise polynomials of degree 1. This pair

of elements achieves the lowest computation cost, but it does not satisfy the inf-sup condition. Therefore, we employ a stabilization technique, such as the Galerkin least squares (GLS) type [21, 27] or the penalty type [12].

(ii) Slip boundary conditions contain two types of boundary conditions: conditions on normal components of the velocity are classified as 'essential boundary conditions', and conditions on the shear stress as 'natural boundary conditions'. To impose a Dirichlet boundary condition (a type of essential boundary conditions), a type of penalty method [25] is used, in which diagonal elements of the stiffness matrices are replaced by a large number. This method is not directly applicable to the slip boundary conditions, nor consistent with the framework of the finite element method.

(iii) It is necessary to eliminate rigid body rotations and pressure lifting from the solution of the Stokes problem. In conventional finite element codes, motions at some nodes are fixed to eliminate rigid body rotations, and a temporary solution is obtained [5]. To adjust the solution, another procedure that consists of restoring data from the kernel of the stiffness matrix is necessary. This procedure requires a condition on the kernel of the stiffness matrix. Unfortunately, in the case of the Stokes equations in a spherical shell domain with slip boundary conditions, this condition is not satisfied due to perturbations caused by the finite element approximation.

For remedies of (ii) and (iii), we introduce a unified procedure to satisfy the essential boundary conditions (e.g., a part of slip boundary conditions) and linear constraints (e.g., rigid body rotations and pressure lifting). A finite element solution is found in a subspace whose functions satisfy the essential boundary conditions and linear constraints. We call the subspace the 'solution space'. The finite element equation is expressed by the discretized variational form in the solution space. This equation is rewritten in a matrix form using the stiffness matrix, which is defined independently on the all finite element nodes of the boundary conditions, and an orthogonal projection onto the solution space. The linear system obtained is solved in Krylov subspaces, which are included in the solution space.

(iv) A stiffness matrix obtained by a finite element discretization is symmetric, although it contains negative eigenvalues and is indefinite. The conjugate gradient (CG) method [26] is known as the simplest method of the Krylov subspace methods that are suitable for solving linear equations with large-scale sparse matrices. The CG method is commonly used for positive definite matrices, but it is not used for indefinite matrices due to the possibility of 'breakdown'. The SYMMLQ or MINRES methods [34] with positive definite preconditioners or the generalized minimal residual (GMRES) method [40] with some preconditioners can be used. However, the GMRES method is designed for unsymmetric matrices and requires large memory storage.

We verify that when a breakdown does not occur, the preconditioned CG (PCG) method can find an appropriate solution in a Krylov subspace. From this investigation, we can employ an indefinite symmetric matrix as a preconditioner, which

improves the convergence of the CG method.

(v) To reduce memory requirements, some computational techniques were developed for domains having some form of symmetry or periodicity. When a domain has a periodicity, Zienkiewicz [51] reduced the problem to one in a unit subdomain, by assuming a periodic solution. When a domain has a symmetry, Bossavit [8] and Bonnet [9] developed a strategy without assuming symmetry of the solution. In their works, a continuous problem is reduced to a set of smaller problems. Here, every sub-problem should have the same type of boundary conditions imposed. This causes a restriction in the domain decomposition. For more drastic memory reduction, we require a more efficient algorithm.

We also use domain decomposition into congruent subdomains to reduce memory requirements. However, we decompose the stiffness matrix ignoring boundary conditions. Therefore, we can use a larger number of subdomains and more flexible decompositions than the previous methods. This idea can be adopted because, in our solver, the stiffness matrix is defined on the all finite element nodes independently of the boundary conditions, and because treatment of the boundary conditions is achieved by the projection operation.

Finally, as the purpose for this thesis, we apply our parallel finite element solver of the Stokes problem to the Earth's mantle convection problem. Temperature-dependent viscosity is considered because of the complex rheological characteristics of the Earth's mantle. Construction of the finite element scheme for the Rayleigh–Bénard equations, the mathematical justification of convergence of the scheme, and some numerical results have been reported in our papers [43, 46, 47]. To treat temperature-dependent viscosity, we extend the Stokes solver to the linear Stokes equations with variable viscosity. Using this solver, we can implement our scheme on parallel computers and obtain numerical results.

Organization of the thesis

In Chapter 1, stabilized finite element methods for the Stokes equations are reviewed. First, we describe an abstract framework of the finite element method and present a finite element equation in a discretized variational form. Second, we show stabilized finite element methods of the GLS type and the penalty type with finite element spaces where Dirichlet boundary conditions or slip boundary conditions are imposed.

In Chapter 2, we present a parallel solver for the Stokes equations. We construct the solver using the P1 finite element approximation, a finite element equation with the orthogonal projection onto the solution space, the preconditioned conjugate gradient method for indefinite symmetric matrices, and domain decomposition into congruent subdomains. Some numerical results are reported to show advantages of the developed parallel Stokes solver.

In Chapter 3, we show an application of our parallel Stokes solver to the Earth's mantle convection problem. We review mathematical modeling of Earth's mantle convection and present a finite element scheme with stabilization techniques.

An extension of the Stokes solver from constant viscosity to variable viscosity is discussed. Numerical results of the evolution problem of the Earth's mantle convection are shown. Finally, the efficiencies of the parallel Stokes solver are presented.

Chapter 1

Stabilized Finite Element Method for the Stokes Equations

1.1 Introduction

In this chapter, we review stabilized finite element methods for the Stokes equations. The Stokes equations describe the motion of an incompressible fluid with slow flow speed. The equations are treated with a weak form of the saddle-point type. Here, an energy functional is minimized under the constraint of incompressibility. The key point to ensure the existence and uniqueness of a solution of the Stokes problem in the weak form is the so-called inf-sup condition [23, 13, 10]. Standard mixed finite elements are used to discretize the weak form of the saddle-point type, where P2/P1 or iso-P2 P1/P1 elements are commonly used. Here, velocity and pressure variables are approximated by P2 element or iso-P2 P1 element, and by P1 element, respectively. In the case of P2 element (piecewise quadratic element), a function is approximated by a polynomial of degree 2 in every simplex element, and in the case of P1 element (piecewise linear element) by a polynomial of degree 1. An iso-P2 P1 element is equal to a P1 element with half the size of simplexes and has the same degrees of freedom as a P2 element. Such a combination of elements P2/P1 or iso-P2 P1/P1 satisfies the inf-sup condition that is independent of the mesh size [23, 13].

Because we deal with 3-D problems, we should be attentive to computation costs. Stiffness matrices generated using P2 element or iso-P2 P1 element for the velocity have many nonzero components. This follows from the fact that ten nodes are considered in the case of a P2 element or an iso-P2 P1 element, whereas four nodes are used in the case of a P1 element. Therefore, there exist many nonzero entries in each row of the stiffness matrix. Consequently, the stiffness matrices require large memory storage. Construction and operations of the stiffness matrices require many arithmetic operations. We employ the same interpolation for both the velocity and pressure by a P1/P1 element. However, it is well known that a P1/P1 combination does not satisfy the inf-sup condition. Therefore, we must use

a stabilization technique of the Galerkin least squares (GLS) type [21, 27] or the penalty type [12]. In the GLS method, properly weighted least-squares terms of the strong form of the differential equations are added to the original weak form. In the penalty method, additional terms, similar to those in the GLS method are added. The weight in the stabilizing terms is set to be proportional to squares of the mesh size. With the stabilizing terms, the finite element equation concerning both the velocity and pressure satisfies the inf-sup condition, and an error estimate of the finite element solution is obtained.

It is also important to consider a numerical procedure to solve the finite element equations. A finite element equation is expressed by a discretized variational formulation in a subspace of the finite element space with total degrees of freedom. The variational equation consists of a bilinear form that satisfies the inf-sup condition. Essential boundary conditions including Dirichlet boundary conditions, and some linear constraints including elimination of the rigid body rotations etc., are built into the subspace. We call this subspace the 'solution space'. A stiffness matrix is defined from the bilinear form with total degrees of freedom. In general, the stiffness matrix may be singular, even if the existence and uniqueness of the finite element solution is guaranteed by satisfaction of the inf-sup condition. In conventional codes of finite element methods, a regular coefficient matrix of the linear system is obtained by fixing some data on finite element nodes and removing some rows and columns of the stiffness matrix [5]. Then, the solution is adjusted by restoring data from the kernel of the stiffness matrix. This procedure is feasible under an assumption that the dimension of the image of the stiffness matrix is equal to the dimension of the solution space. In general cases, this assumption may not be satisfied. For example, perturbations due to approximations by the finite elements cause the inconvenience.

We derive a linear system by rewriting the discretized variational equation with the stiffness matrix and an orthogonal projection onto the solution space. We use a Krylov subspace method to solve this linear system. Krylov subspaces are generated with the stiffness matrix multiplied by the projection matrices from both sides and an initial residual, and are included in the solution space. We define regularity of a matrix in a subspace to ensure the existence and uniqueness of a solution of the finite element equation in the solution space. Using this formulation, we can avoid singular linear systems that cause some difficulties in Krylov subspace methods [32, 49].

Organization of this chapter

In Section 1.2, we describe the Stokes problems with Dirichlet boundary conditions or slip boundary conditions. In Section 1.3, we first review an abstract formulation of the finite element method. Second, we show an approximation property of the finite element solution. Third, we present a discretized equation expressed by a stiffness matrix with an orthogonal projection onto the solution space. In Section 1.4, we show the existence and uniqueness of a solution of the Stokes problem with either

boundary condition of two different types. In Section 1.5, we review the stabilization techniques of the GLS type and the penalty type for the Stokes problem. Following Franca–Stenberg [22], we show error estimates for both schemes.

Notations of vector and matrix

The kth component of the vector $\vec{u} \in \mathbb{R}^N$ is denoted by $[\vec{u}]_k$, where N is a positive integer, and the component at the (k,l) entry of matrix $A \in \mathbb{R}^{N\times N}$ is denoted by $[A]_{kl}$. We use the notation $(\vec{u}, \vec{v})_{\mathbb{R}^N}$ to represent the inner product of vectors $\vec{u}, \vec{v} \in \mathbb{R}^N$. We also use the standard notation $u \cdot v$ to represent the inner product of $\mathbb{R}^3$-valued functions u and v.

1.2 Stokes problem in 3-D domain with Dirichlet or slip boundary conditions

Let Ω be a bounded domain in $\mathbb{R}^3$ and Γ be its boundary. Let n be the unit outer normal to the boundary. We consider the Stokes equations of velocity $u = (u_1, u_2, u_3)$ and pressure p satisfying

$$-2\nabla \cdot D(u) + \nabla p = f \qquad \text{in } \Omega\,, \tag{1.1a}$$

$$\nabla \cdot u = 0 \qquad \text{in } \Omega\,. \tag{1.1b}$$

Here $D(u)$ is the strain rate tensor

$$[D(u)]_{ij} := \frac{1}{2}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right)$$

and

$$[\nabla \cdot D(u)]_i := \sum_{j=1}^{3} \frac{\partial}{\partial x_j}[D(u)]_{ij}\,.$$

We deal with the Stokes problem with either kind of the following boundary conditions.

Problem 1.1 (*Dirichlet Problem*) Let Ω be a polyhedral bounded domain in $\mathbb{R}^3$. We consider the Stokes equations (1.1a) and (1.1b) in Ω with Dirichlet boundary conditions imposed on Γ,

$$u = g \tag{1.2}$$

where g is a given $\mathbb{R}^3$-valued function on Γ with

$$\int_\Gamma g \cdot n \, ds = 0\,.$$

Problem 1.2 (*Slip Problem*) Let Ω be a ball

$$\Omega := \{x \in \mathbb{R}^3\,;\, |x| < R_2\},$$

or a spherical shell domain

$$\Omega := \{x \in \mathbb{R}^3\,;\, R_1 < |x| < R_2\},$$

where R_1 and R_2 are positive constants. We consider the Stokes equations $(1.1a)$ and $(1.1b)$ in Ω with the slip boundary conditions imposed on Γ,

$$u \cdot n = g_n\,, \tag{1.3a}$$
$$n \times (D(u)n \times n) = g_t\,, \tag{1.3b}$$

where g_n is a given real-valued function on Γ with

$$\int_\Gamma g_n\, ds = 0\,,$$

and g_t is a given $\mathbb{R}^3$-valued function on Γ with

$$g_t \cdot n = 0\,.$$

Our aim in this chapter is to review the framework of stabilized finite element methods and linear solvers for the Stokes problem. For this purpose, we assume that Ω is a polyhedral domain in Problem 1.1 to avoid complexities from approximation of the smooth domain by polyhedral domains, while we treat a spherical shell domain whose boundary is smooth in Problem 1.2. As in Problem 1.2, we can consider general domains in case of Dirichlet boundary conditions [45].

1.3 Abstract framework of finite element method

1.3.1 A bilinear form and finite element equations

Let Z be a real Hilbert space with an inner product $(\cdot,\cdot)_Z$ and its corresponding norm $||\cdot||_Z$. Let Y be a closed subspace of Z. Let $\alpha(\cdot,\cdot)$ be a continuous bilinear form on Z .

Assumption 1.1 *The bilinear form $\alpha(\cdot,\cdot)$ satisfies the inf-sup condition on Y, i.e., there exists a positive constant α_0 such that*

$$\inf_{u\in Y}\sup_{v\in Y}\frac{\alpha(u,v)}{||u||_Z||v||_Z} \geq \alpha_0\,, \tag{1.4}$$

and

$$\inf_{v\in Y}\sup_{u\in Y}\frac{\alpha(u,v)}{||u||_Z||v||_Z} \geq \alpha_0\,. \tag{1.5}$$

We consider a problem to find $u \in Y$ satisfying

$$\alpha(u, v) = (f, v)_Z \tag{1.6}$$

for all $v \in Y$, where $f \in Z$.

Lemma 1.1 *Under Assumption 1.1, there exists a unique solution of* (1.6). Several proofs are found in [10, 13, 36].

Let $h > 0$ be a discretization parameter corresponding to a mesh size. For each h, let $Z_h \subset Z$ and $Y_h \subset Y$ be finite-dimensional subspaces with $Y_h \subset Z_h$. We suppose that $\alpha_h(\cdot, \cdot)$ is a continuous bilinear form on Z satisfying the following property.

Assumption 1.2 *There exists a positive constant α_1 such that for all positive number h*

$$\inf_{u_h \in Y_h} \sup_{v_h \in Y_h} \frac{\alpha_h(u_h, v_h)}{||u_h||_Z ||v_h||_Z} \geq \alpha_1 \,. \tag{1.7}$$

We call (1.7) '*stability inequality*'.

We consider a finite element problem to find $u_h \in Y_h$ satisfying

$$\alpha_h(u_h, v_h) = (f, v_h)_Z \,, \tag{1.8}$$

for all $v_h \in Y_h$, where $f \in Z$.

Lemma 1.2 *Under Assumption 1.2, there exists a unique solution of* (1.8).

Proof. Let A be a linear transformation from Y_h to Y_h defined by

$$(Au_h, v_h)_Z = \alpha_h(u_h, v_h),$$

for all u_h, $v_h \in Y_h$. The injectivity of A follows from (1.7). Since Y_h is a finite-dimensional vector space, the injectivity leads to the surjectivity of A. □

Remark 1.1 Under Assumption 1.2, we also have

$$\inf_{v_h \in Y_h} \sup_{u_h \in Y_h} \frac{\alpha_h(u_h, v_h)}{||u_h||_Z ||v_h||_Z} \geq \alpha_1 \,.$$

We make an assumption on the consistency of the finite element scheme:

Assumption 1.3 *Let $u \in Y$ and $u_h \in Y_h$ be solutions of the variational problem* (1.6) *and of the finite element problem* (1.8)*, respectively. The bilinear form $\alpha_h(\cdot, \cdot)$ satisfies*

$$\alpha_h(u - u_h, v_h) = 0$$

for all $v_h \in Y_h$.

We make another assumption.

Assumption 1.4 *There exists a positive constant A_1 such that for all positive number h, $u \in Y$, and $v_h \in Y_h$*

$$\alpha_h(u, v_h) \le A_1 ||u||_Z ||v_h||_Z \,.$$

We have an extension of Céa's lemma [15, 10].

Theorem 1.1 *Let $u \in Y$ and $u_h \in Y_h$ be solutions of the variational problem* (1.6) *and the finite element problem* (1.8)*, respectively.*
(i) *Under Assumptions 1.1, and 1.2, the following estimate holds:*

$$||u-u_h||_Z \le \inf_{v_h \in Y_h} \left(||u - v_h||_Z + \frac{1}{\alpha_1} \sup_{w_h \in Y_h} \frac{\alpha_h(u - v_h, w_h)}{||w_h||_Z} \right) + \frac{1}{\alpha_1} \sup_{w_h \in Y_h} \frac{\alpha_h(u - u_h, w_h)}{||w_h||_Z} \,.$$

(ii) *Under Assumptions 1.1, 1.2, and 1.3, the following estimate holds:*

$$||u - u_h||_Z \le \inf_{v_h \in Y_h} \left(||u - v_h||_Z + \frac{1}{\alpha_1} \sup_{w_h \in Y_h} \frac{\alpha_h(u - v_h, w_h)}{||w_h||_Z} \right) \,.$$

(iii) *Under Assumptions 1.1, 1.2, 1.3, and 1.4, the following estimate holds:*

$$||u - u_h||_Z \le (1 + \frac{A_1}{\alpha_1}) \inf_{v_h \in Y_h} ||u - v_h||_Z \,.$$

Proof. Let $v_h \in Y_h$. Then we have

$$\begin{aligned} ||u - u_h||_Z &\le ||u - v_h||_Z + ||u_h - v_h||_Z \\ &\le ||u - v_h||_Z + \frac{1}{\alpha_1} \sup_{w_h \in Y_h} \frac{\alpha_h(u_h - v_h, w_h)}{||w_h||_Z} \\ &\le ||u - v_h||_Z + \frac{1}{\alpha_1} \sup_{w_h \in Y_h} \frac{\alpha_h(u - v_h, w_h)}{||w_h||_Z} + \frac{1}{\alpha_1} \sup_{w_h \in Y_h} \frac{\alpha_h(u - u_h, w_h)}{||w_h||_Z} \,. \end{aligned}$$

This inequality leads to the desired results. □

Let $g \in Z$ and $g_h \in Z_h$. We define a subset $Y(g) \subset Z$ and an affine finite element space $Y_h(g_h) \subset Z_h$ by

$$\begin{aligned} Y(g) &:= \{u \in Z \,;\, u = v + g,\ v \in Y\} \,, \\ Y_h(g_h) &:= \{u_h \in Z_h \,;\, u_h = v_h + g_h,\ v_h \in Y_h\} \,. \end{aligned}$$

Let $f \in Z$. We consider a problem to find $u \in Y(g)$ satisfying

$$\alpha(u, v) = (f, v)_Z \tag{1.9}$$

for all $v \in Y$.

Lemma 1.3 *Under Assumption 1.1, there exists a unique solution of* (1.9).
The result follows from the linearity of the bilinear form $\alpha(\cdot, \cdot)$ and Lemma 1.1.

Corresponding finite element problem is to find $u_h \in Y_h(g_h)$ satisfying

$$\alpha_h(u_h, v_h) = (f, v_h)_Z \tag{1.10}$$

for all $v_h \in Y_h$.

Lemma 1.4 *Under Assumption 1.2, there exists a unique solution of* (1.10).

We make an assumption on the consistency in the same manner as Assumption 1.3:

Assumption 1.5 *Let $u \in Y(g)$ be a solution of the variational problem* (1.9) *and $u_h \in Y_h(g_h)$ be of the finite element problem* (1.10)*, respectively. The bilinear form $\alpha_h(\cdot, \cdot)$ satisfies*

$$\alpha_h(u - u_h, v_h) = 0$$

for all $v_h \in Y_h$.

Theorem 1.2 *Let $u \in Y(g)$ and $u_h \in Y_h(g_h)$ be solutions of the variational problem* (1.9) *and of the finite element problem* (1.10)*, respectively.*
(i) *Under Assumptions 1.1 and 1.2, the following estimate holds:*

$$\begin{aligned} &||u - u_h||_Z \\ &\leq \inf_{v_h \in Y_h(g_h)} \left(||u - v_h||_Z + \frac{1}{\alpha_1} \sup_{w_h \in Y_h} \frac{\alpha_h(u - v_h, w_h)}{||w_h||_Z} \right) + \frac{1}{\alpha_1} \sup_{w_h \in Y_h} \frac{\alpha_h(u - u_h, w_h)}{||w_h||_Z} . \end{aligned} \tag{1.11}$$

(ii) *Under Assumptions 1.1, 1.2, and 1.5, the following estimate holds:*

$$||u - u_h||_Z \leq \inf_{v_h \in Y_h(g_h)} \left(||u - v_h||_Z + \frac{1}{\alpha_1} \sup_{w_h \in Y_h} \frac{\alpha_h(u - v_h, w_h)}{||w_h||_Z} \right) . \tag{1.12}$$

Proof. Let $v_h \in Y_h(g_h)$. Since $u_h - v_h \in Y_h$, we obtain desired estimates in the same manner as Theorem 1.1. □

These estimates are used to establish error estimates of stabilized finite element approximations to the Stokes problem described in Section 1.5, Theorem 1.4.

1.3.2 Matrix formulation of finite element equation

Let $n_Z := \dim Z_h$. We define an index set $\Lambda_Z := \{1, 2, \ldots, n_Z\}$. Let $\{\varphi_\alpha\}_{\alpha \in \Lambda_Z}$ be a finite element basis of Z_h. We define a vector space $\vec{Z}$ by $\vec{Z} := \mathbb{R}^{n_Z}$. A finite element function $v_h \in Z_h$ is expressed by a vector $\vec{v} \in \vec{Z}$ and the finite element basis $\{\varphi_\alpha\}_{\alpha \in \Lambda_Z}$:

$$v_h = \sum_{\alpha \in \Lambda_Z} [\vec{v}]_\alpha \varphi_\alpha .$$

We prepare the following vector space and affine space:

$$\vec{Y} := \{\vec{y} \in \vec{Z}\,;\, \sum_{\alpha\in\Lambda_Z} [\vec{y}]_\alpha \varphi_\alpha \in Y_h\}\,, \tag{1.13}$$

$$\vec{Y}(g) := \{\vec{y} \in \vec{Z}\,;\, \sum_{\alpha\in\Lambda_Z} [\vec{y}]_\alpha \varphi_\alpha \in Y_h(g)\}\,. \tag{1.14}$$

Let $P_{\vec{Y}}$ be the orthogonal projection of $\vec{Z}$ onto $\vec{Y}$,

$$(P_{\vec{Y}}\vec{u},\, \vec{v})_{\mathbb{R}^{n_Z}} = (\vec{u},\, \vec{v})_{\mathbb{R}^{n_Z}}$$

for all $\vec{v} \in \vec{Y}$. Let $\mathcal{A} \in \mathbb{R}^{n_Z\times n_Z}$ be a stiffness matrix and $\vec{f} \in \mathbb{R}^{n_Z}$ be a load vector defined by

$$[\mathcal{A}]_{\alpha\,\beta} := \alpha_h(\varphi_\beta, \varphi_\alpha) \qquad (\alpha, \beta \in \Lambda_Z)\,, \tag{1.15}$$

$$[\vec{f}\,]_\alpha := (f, \varphi_\alpha)_Z \qquad (\alpha \in \Lambda_Z)\,. \tag{1.16}$$

The finite element equation (1.10) is expressed by a matrix formulation, which is to find $\vec{u} \in \vec{Y}(g)$ satisfying

$$(\mathcal{A}\vec{u},\, \vec{v})_{\mathbb{R}^{n_Z}} = (\vec{f},\, \vec{v})_{\mathbb{R}^{n_Z}} \tag{1.17}$$

for all $\vec{v} \in \vec{Y}$. Since (1.10) has a unique solution, (1.17) has a unique solution and the solution is obtained by the following way. Let $\vec{u}_g \in \vec{Y}(g)$. We solve a problem to find $\vec{u}_0 \in \vec{Y}$ satisfying

$$(\mathcal{A}\vec{u}_0,\, \vec{v})_{\mathbb{R}^{n_Z}} = (\vec{f} - \mathcal{A}\vec{u}_g,\, \vec{v})_{\mathbb{R}^{n_Z}} \tag{1.18}$$

for all $\vec{v} \in \vec{Y}$. Then $\vec{u}_0 + \vec{u}_g \in \vec{Y}(g)$ and it satisfies (1.17). The equation (1.18) can be written with the projection $P_{\vec{Y}}$ as follows,

$$P_{\vec{Y}}\mathcal{A}P_{\vec{Y}}\vec{u}_0 = P_{\vec{Y}}(\vec{f} - \mathcal{A}\,\vec{u}_g)\,. \tag{1.19}$$

1.3.3 Regularity of a matrix in a subspace

We define regularity of a matrix in a subspace. Let A be an $N \times N$ real matrix.

Definition 1.1 *Let $\vec{V}$ be a subspace of $\mathbb{R}^N$ with $\dim\vec{V} = m$. We define an $m \times m$ matrix A_m by*

$$[A_m]_{i\,j} := (A\vec{v}^{(j)},\, \vec{v}^{(i)})_{\mathbb{R}^N} \quad (1 \le i, j \le m),$$

where $\{\vec{v}^{(j)}\}_{j=1}^m$ is a basis of $\vec{V}$. A is called regular in $\vec{V}$ if A_m is regular.

Let A be regular in $\vec{V}$. We consider a variational problem to find $\vec{x} \in \vec{V}$ satisfying

$$(A\vec{x},\, \vec{y})_{\mathbb{R}^N} = (\vec{b},\, \vec{y})_{\mathbb{R}^N} \tag{1.20}$$

for all $\vec{y} \in \vec{V}$, where $\vec{b} \in \mathbb{R}^N$. Let $P_{\vec{V}}$ be the orthogonal projection of $\mathbb{R}^N$ onto $\vec{V}$,

$$(P_{\vec{V}}\vec{u},\, \vec{v})_{\mathbb{R}^N} = (\vec{u},\, \vec{v})_{\mathbb{R}^N} \quad \text{for all } \vec{v} \in \vec{V}\,.$$

The variational problem (1.20) is equivalent to the problem to find $\vec{x} \in \vec{V}$ satisfying

$$P_{\vec{V}} A P_{\vec{V}} \vec{x} = P_{\vec{V}} \vec{b}. \tag{1.21}$$

Proposition 1.1 *Suppose that A be regular in $\vec{V}$. Then the variational problem* (1.20) *has the unique solution that is expressed by $\vec{x} = (P_{\vec{V}} A P_{\vec{V}})^{\dagger} P_{\vec{V}} \vec{b}$. Here $M^{\dagger}$ is the pseudo-inverse (the Moore–Penrose inverse)* [24] *of a matrix M.*

1.3.4 Solvers for finite element equation

Since the finite element equation (1.10) has a unique solution, the coefficient matrix of the linear system (1.19), $P_{\vec{Y}} \mathcal{A} P_{\vec{Y}}$ is regular in $\vec{Y}$ and $(P_{\vec{Y}} \mathcal{A} P_{\vec{Y}})^{\dagger} P_{\vec{Y}} (\vec{f} - \mathcal{A}\,\vec{u}_g) + \vec{u}_g$ is the solution of (1.17). We will make some comments on solvers of the linear system (1.19).

Krylov subspace methods

We note that the linear system (1.19) is considered in the space $\vec{Y}$ and the matrix $P_{\vec{Y}} \mathcal{A} P_{\vec{Y}}$ is regular in $\vec{Y}$. Krylov subspace methods are suitable for solving the linear system. The main procedure of the Krylov subspace methods consists of multiplications of the matrix by the vector. To obtain a product of the matrix $P_{\vec{Y}} \mathcal{A} P_{\vec{Y}}$ and a vector, it is sufficient to know the values of the stiffness matrix $\mathcal{A}$ and the way of operation of the projection $P_{\vec{Y}}$. We need not to know the exact values of all entries of the matrix $P_{\vec{Y}} \mathcal{A} P_{\vec{Y}}$. In addition, we also need not to know any information on the kernel of the stiffness matrix $\mathcal{A}$. When the matrix $\mathcal{A}$ is symmetric, the conjugate gradient (CG) method [26] is used. Otherwise, the bi-conjugate gradient (BiCG) method [20], the generalized minimal residual (GMRES) method [40], or the generalized conjugate residual (GCR) method [19] is used. We employ the CG method to solve finite element equations of the Stokes problems. Details of the CG method for a symmetric matrix are discussed in Section 2.3.

A solver by fixing some data on finite element nodes

In conventional codes of finite element methods [5], a regular coefficient matrix of the linear system is obtained by fixing some data on finite element nodes and removing some rows and columns of the stiffness matrix. Then, the solution is adjusted by restoring data from the kernel of the stiffness matrix. This procedure can be summarized as a general method to find a solution of the problem (1.18). We will precisely show the method for the problem with a symmetric matrix $\mathcal{A}$ in

Appendix A.1. This method is feasible under an assumption that the dimension of the image of the stiffness matrix is equal to the dimension of the solution space. We will show this condition is not satisfied in the Stokes equations in a spherical shell domain with slip boundary conditions imposed: the *Slip Problem*, in Appendix A.2. This is the reason why we use a Krylov subspace method.

1.4 A weak formulation of the Stokes problem

Bilinear forms and weak formulations

Let $H^m(\Omega)$ be the standard Sobolev space [15, 23] with a nonnegative integer m, and its norm is denoted by $||\cdot||_m$. When $m = 0$, it is written as $L^2(\Omega)$. The symbols $L^2(\Omega)^3$ and $H^1(\Omega)^3$ denote the $\mathbb{R}^3$-valued function spaces whose components belong to $L^2(\Omega)$ and $H^1(\Omega)$, respectively. We use the same notation $(\cdot, \cdot\,;\,\Omega)$ to represent the inner products in both $L^2(\Omega)$ and $L^2(\Omega)^3$, i.e., for $u, v \in L^2(\Omega)^3$ and $p, q \in L^2(\Omega)$,

$$(u, v\,;\,\Omega) := \int_\Omega u \cdot v \, dx\,,$$
$$(p, q\,;\,\Omega) := \int_\Omega p\, q \, dx\,.$$

We also use the same notation for the norm $||\cdot||_0$ in scalar- and vector-valued L^2-functions. We prepare the following bilinear forms: for $u,\ v \in H^1(\Omega)^3$, $q \in L^2(\Omega)$,

$$a(u, v\,;\,\Omega) := 2\int_\Omega D(u) : D(v)\, dx,$$
$$b(v, q\,;\,\Omega) := -\int_\Omega \nabla \cdot v\, q \, dx\,,$$

where

$$D(u) : D(v) := \sum_{1\le i,j\le 3} [D(u)]_{ij}[D(v)]_{ij}\,.$$

We omit Ω if there is no confusion, e.g., we use $(\cdot, \cdot)$ in place of $(\cdot, \cdot, \Omega)$ and $a(\cdot, \cdot)$ in place of $a(\cdot, \cdot\,;\,\Omega)$. We prepare the following Hilbert spaces for both problems:

$$X := H^1(\Omega)^3,$$
$$Q := L^2_0(\Omega) = \left\{q \in L^2(\Omega)\ ;\ (q,\,1) = 0\right\}\,. \tag{1.22}$$

We use $L^2(\Omega)$-norm for Q.

In the *Dirichlet Problem* we define a subset $V(g)$ and a space V by

$$V(g) := \{v \in X\ ;\ v = g \text{ on } \Gamma\}\,, \tag{1.23}$$
$$V := V(0)\,. \tag{1.24}$$

We note that V is equivalent to the space $H^1_0(\Omega)^3$. We use $H^1(\Omega)^3$-norm for V.

A variational formulation for Problem 1.1 is to find $(u, p) \in V(g) \times Q$ satisfying

$$a(u, v) + b(v, p) = (f, v), \tag{1.25a}$$
$$b(u, q) = 0 \tag{1.25b}$$

for all $(v, q) \in V \times Q$.

We note that the Dirichlet boundary conditions (1.2) is a type of essential boundary conditions, and it characterizes the subset $V(g)$.

In the *Slip Problem* we define subsets $W(g_n)$ and $V(g_n)$ and spaces W and V by

$$W(g_n) := \{v \in X \ ; \ v \cdot n = g_n \text{ on } \Gamma\},$$
$$W := W(0),$$
$$V(g_n) := \left\{v \in W(g_n) \ ; \ (v, v^{(i)}) = 0 \ (i = 1, 2, 3)\right\}, \tag{1.26}$$
$$V := V(0), \tag{1.27}$$

where $v^{(i)}(x) := e^{(i)} \times x$, $i = 1, 2, 3$. Here $e^{(i)}$ is the unit vector to the x_i-direction. We use $H^1(\Omega)^3$-norm for V.

A variational formulation for Problem 1.2 is to find $(u, p) \in V(g_n) \times Q$ satisfying

$$a(u, v) + b(v, p) = (f, v) + 2\int_\Gamma g_t \cdot v \, ds, \tag{1.28a}$$
$$b(u, q) = 0 \tag{1.28b}$$

for all $(v, q) \in V \times Q$.

We note that $(1.3a)$ is a type of essential boundary conditions, and it characterizes the subset $V(g_n)$, and that $(1.3b)$ is a type of natural boundary conditions, and it is included in the right-hand term of $(1.28a)$.

Existence and uniqueness of the solution

In the *Dirichlet Problem* we have two inequalities that ensure the existence and uniqueness of a solution of the Stokes problem. We assumed that in Problem 1.1, Ω is the polyhedral bounded domain.

Lemma 1.5 *Let V and Q be spaces defined by* (1.24) *and* (1.22)*, respectively.*
(i) *$a(\cdot, \cdot)$ is coercive on V, i.e., there exists a positive constant α_0 such that*

$$\inf_{v \in V} \frac{a(v, v)}{||v||_V^2} \geq \alpha_0.$$

(ii) *$b(\cdot, \cdot)$ satisfies the inf-sup condition on $Q \times V$, i.e., there exists a positive constant β_0 such that*

$$\inf_{q \in Q} \sup_{v \in V} \frac{b(v, q)}{||q||_Q ||v||_V} \geq \beta_0.$$

The coerciveness follows from Korn's inequality [18, 11], i.e., there exists a positive constant c such that

$$\int_\Omega D(v) : D(v)\, dx + ||v||_0^2 \geq c\, ||v||_1^2$$

for all $v \in H^1(\Omega)^3$, and Poincaré's inequality [11]. A proof of the inf-sup condition is found in [23].

In the *Slip Problem* the external force $f \in L^2(\Omega)^3$ and the shear stress $g_t \in H^{1/2}(\Omega)^3$ are assumed to satisfy a constraint,

$$\int_\Omega f \cdot v^{(i)} dx + 2 \int_\Gamma g_t \cdot v^{(i)} ds = 0 \quad (i = 1, 2, 3)\,, \tag{1.29}$$

where $v^{(i)}(x) = e^{(i)} \times x$.

Remark 1.2
(i) $v^{(i)},\ i = 1, 2, 3$, is a rigid body rotation and satisfies $a(v^{(i)}, v^{(i)}) = 0$. The bilinear form a is not coercive on W.
(ii) We have

$$\nabla \cdot v^{(i)} = 0 \quad (i = 1, 2, 3)\,.$$

We have the same results as in Lemma 1.5.

Lemma 1.6 *Let V and Q be spaces defined by* (1.27) *and* (1.22)*, respectively.*
(i) *$a(\cdot, \cdot)$ is coercive on V.*
(ii) *$b(\cdot, \cdot)$ satisfies the inf-sup condition on $Q \times V$.*
Proof. We will prove the coerciveness of $a(\cdot, \cdot)$. Since the subspace $\{w \in W\ ;\ a(w, w) = 0\}$ is equal to the subspace: span$[v^{(1)}, v^{(2)}, v^{(3)}]$, the coerciveness of $a(\cdot, \cdot)$ follows from Korn's inequality [18, 11], i.e., there exists a positive constant c such that

$$a(v, v) \geq c||v||_1^2$$

for all $v \in H^1(\Omega)^3/\mathcal{R}$, where $\mathcal{R}$ is the set of rigid body motions,

$$\mathcal{R} := \{v \in H^1(\Omega)^3\,;\, v(x) = a + b \times x,\ (a, b \in \mathbb{R}^3)\}\,.$$

We will prove the inf-sup condition. Let v be any function in $H_0^1(\Omega)^3$. We decompose v as follows

$$v = v_0 + \sum_{i=1}^{3} c_i v^{(i)}, \qquad c_i = (v, v^{(i)})/(v^{(i)}, v^{(i)})\,.$$

Then we have $v_0 \in V$ and $||v_0||_V \leq c\, ||v||_V$. Noting that $\operatorname{div} v^{(i)} = 0$, we obtain

$$\inf_{q \in Q} \sup_{v \in H_0^1(\Omega)^3} \frac{b(v, q)}{||q||_Q ||v||_V} \leq c \inf_{q \in Q} \sup_{v \in V} \frac{b(v, q)}{||q||_Q ||v||_V}\,.$$

The assertion follows from Lemma 1.5, (ii). □

In both the *Dirichlet Problem* and *Slip Problem*, we define the following bilinear form: for $(u,p),(v,q) \in V \times Q$,

$$A(u,\, p\,;\, v,\, q) := a(u,\, v) + b(v,\, p) + b(u,\, q)\,.$$

From Lemmas 1.5 and 1.6 we obtain the following result.

Lemma 1.7 *There exists a positive constant* α *such that*

$$\inf_{(u,\,p)\in V\times Q} \sup_{(v,\,q)\in V\times Q} \frac{A(u,\, p\,;\, v,\, q)}{||(u,\, p)||_{V\times Q}||(v,\, q)||_{V\times Q}} \geq \alpha\,.$$

Its proof is found in [10].

We note that the bilinear form $A(\cdot\,;\,\cdot)$ is symmetric. The existence and uniqueness of a solution of the Stokes problems follows from Lemma 1.7.

Theorem 1.3
(i) *For every* $f \in L^2(\Omega)^3$ *and* $g \in H^{3/2}(\Gamma)^3$*, there exists a unique solution of* (1.25) *and it satisfies* (1.1) *and* (1.2)*.*
(ii) *For every* $f \in L^2(\Omega)^3$ *and* $g_t \in H^{1/2}(\Gamma)^3$ *satisfying* (1.29)*, and* $g_n \in H^{3/2}(\Gamma)$*, there exists a unique solution of* (1.28) *and it satisfies* (1.1) *and* (1.3)*.*

Here $H^{m-1/2}(\Gamma)$ is a Sobolev space on the boundary to which the trace of a function in $H^m(\Omega)$ belongs.

Remark 1.3 In Theorem 1.3, we consider a smooth solution whose regularity is higher than a weak solution of the Stokes problem with a variational form. Such higher regularity of the solution is required in stabilization techniques for finite element approximations, which will be described in Section 1.5.

1.5 A P1/P1 finite element approximation with stabilization technique of Galerkin least squares type or penalty type

Because our problem is considered in three dimension, we should be attentive to computation costs. We employ an economical element combination, P1/P1, where both the velocity and pressure are approximated by piecewise polynomials of degree 1. Note that the number of degrees of freedom of P1 (linear tetrahedral element) in $\bar{\Omega}$ is smaller than that of P0 (constant tetrahedral element). While the P1/P1 element for the Stokes equations may much reduce memory requirements in comparison with other elements such as P2/P1 element and iso-P2 P1/P1 element, it does not satisfy the inf-sup condition that allows to solve the discretized Stokes problem [23]. We, therefore, employ a stabilization technique of the Galerkin least squares (GLS) type [21, 27] or the penalty type [12]. We note that in case of P1 element, the Douglas–Wang method [17], which is also a stabilized finite element method, is equivalent to the GLS method.

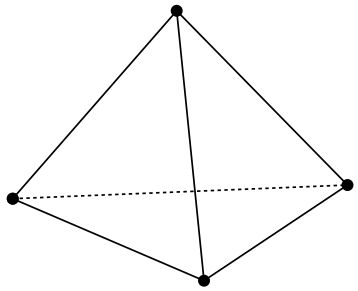

Figure 1.1. Nodes of a P1 element in three dimension

P1 finite element spaces

In the *Dirichlet Problem*, let $\mathcal{T}_h$ be a partition of $\bar{\Omega}$ by tetrahedra, where h is the maximum diameter of tetrahedral elements. In the *Slip Problem*, let Ω_h be a polyhedral approximation to Ω and $\mathcal{T}_h$ be a partition of $\bar{\Omega}_h$ by tetrahedra. We consider a regular family of partitions $\{\mathcal{T}_h\}_{h\downarrow 0}$ [15], i.e., there exists a positive constant σ_0 such that for all $h > 0$ every element $K \in \mathcal{T}_h$ contains a sphere with diameter $\rho_K \geq h_K/\sigma_0$, where h_K is the diameter of K. The boundary of Ω_h is denoted by Γ_h.

Let $\mathcal{L}_1^1(\Omega)$ be the P1 finite element space defined by

$$\mathcal{L}_1^1(\Omega) := \{v \in C^0(\bar{\Omega})\,;\, v|_K \in \mathcal{P}^1(K), K \in \mathcal{T}_h\}\,,$$

where $\mathcal{P}^1$ is the set of polynomials of degree 1. Degrees of freedom of P1 element are numbers of the vertices of tetrahedra. We call each vertex of the tetrahedra '*node*'. Each P1 finite element function $\psi^{(k)}(k = 1, \ldots, 4)$ in a element $K \in \mathcal{T}_h$ is expressed as

$$\psi^{(k)}(x) = a_0^{(k)} + a_1^{(k)}[x]_1 + a_2^{(k)}[x]_2 + a_3^{(k)}[x]_3\,,$$

and satisfies that

$$\psi^{(k)}(P_l) = \delta_{k\,l},$$

where $\{P_l\}_{l=1}^4$ are nodes of K. Figure 1.1 shows nodes of the P1 element in three dimension.

In the *Dirichlet Problem* we introduce finite element spaces X_h, V_h, and Q_h corresponding to X, V, and Q, respectively,

$$\begin{aligned}
S_h &:= \mathcal{L}_1^1(\Omega)\,, \\
X_h &:= {S_h}^3\,, \\
V_h(g) &:= \{v_h \in X_h\ ;\ v_h(P) = g(P)\ (\forall P)\}\,, && (1.30)\\
V_h &:= V_h(0)\,, && (1.31)\\
Q_h &:= \{q_h \in S_h\ ;\ (q_h, 1) = 0\}\ , && (1.32)
\end{aligned}$$

where P stands for a node on Γ.

Lemma 1.8 *Let V_h and Q_h be finite element spaces defined by* (1.31) *and* (1.32)*, respectively. If $u \in H^2(\Omega)^3 \cap H_0^1(\Omega)^3$ and $p \in H^1(\Omega) \cap L_0^2(\Omega)$, interpolation errors by V_h and Q_h are estimated by*

$$\inf_{v_h \in V_h} ||u - v_h||_1 \leq c\, h\, ||u||_2\,,$$
$$\inf_{q_h \in Q_h} ||p - q_h||_0 \leq c\, h\, ||p||_1\,,$$

where each c is a constant independent of h. Here $L_0^2(\Omega)$ is defined in (1.22)*.*

In the *Slip Problem* we use finite element spaces $\tilde{W}_h$, $\tilde{V}_h$, and $\tilde{Q}_h$ corresponding to W, V, and Q, respectively,

$$\begin{aligned}
\tilde{S}_h &:= \mathcal{L}_1^1(\Omega_h)\,,\\
\tilde{X}_h &:= \tilde{S}_h^3\,,\\
\tilde{W}_h(g_n) &:= \left\{ v_h \in \tilde{X}_h \ ;\ (v_h \cdot n_\Omega)(P) = g_n(P)\ (\forall P) \right\},\\
\tilde{W}_h &:= \tilde{W}_h(0)\,,\\
\tilde{V}_h(g_n) &:= \left\{ v_h \in W_h(g_n) \ ;\ (v_h, v^{(i)})_h = 0\ (i = 1,2,3) \right\}, \qquad (1.33)\\
\tilde{V}_h &:= \tilde{V}_h(0)\,, \qquad (1.34)\\
\tilde{Q}_h &:= \left\{ q_h \in \tilde{S}_h \ ;\ (q_h, 1)_h = 0 \right\}, \qquad (1.35)
\end{aligned}$$

where P stands for a node on Γ_h, n_Ω is the unit outer normal to Γ. Since we use the P1 element, every node P on Γ_h is on Γ. Here every integral over Ω is replaced by that over Ω_h. We express such integrals by adding the subscript h. For example, we use $a_h(\cdot,\cdot)$ in place of $a(\cdot,\cdot\,;\,\Omega_h)$ and $(\cdot,\cdot)_h$ in place of $(\cdot,\cdot\,;\,\Omega_h)$.

We note that the rigid body rotation $v^{(i)}$, $i = 1, 2, 3$, belongs to $\tilde{W}_h$.

Remark 1.4 Verfürth [48] introduced a Lagrange multiplier to impose the slip boundary conditions. We simply impose the slip boundary conditions at nodes by using the normal vector to the exact domain, which does not deteriorate the convergence rate.

To describe stabilized finite element schemes we prepare a bilinear form for $p,\, q \in H^1(\Omega)$,

$$d(p,\, q) := \sum_{K \in \mathcal{T}_h} h_K^2 (\nabla p,\, \nabla q\,;\, K)\,,$$

where h_K is the diameter of K. We recall that $(\cdot,\cdot\,;\,K)$ is the L^2-inner product on element K. We define a semi-norm $|\cdot|_h$ for $H^1(\Omega)$ by

$$|q|_h := d(q,\, q)^{1/2}\,.$$

Lemma 1.9
(i) *We have the following estimate:*

$$|q_h|_h \le c||q_h||_0 \tag{1.36}$$

for $q_h \in S_h$, where c is a constant independent of h.
(ii) *If $p \in H^1(\Omega)$, an interpolation error is estimated by*

$$\inf_{q_h \in S_h} |p - q_h|_h \le c\, h\, ||p||_1\,, \tag{1.37}$$

where c is a constant independent of h.

Stabilized finite element methods

We define the following bilinear forms: for $(u_h, p_h),\ (v_h, q_h) \in X_h \times S_h$,

$$\begin{aligned} A_\delta^{\mathrm{GLS}}(u_h,\, p_h\,;\, v_h,\, q_h) &:= a(u_h, v_h) + b(v_h, p_h) + b(u_h, q_h) \\ &\quad - \delta \sum_{K \in \mathcal{T}_h} h_K^2 \left(2\nabla \cdot D(u_h) + \nabla p_h,\ 2\nabla \cdot D(v_h) + \nabla q_h\,;\, K \right), \\ A_\delta^{\mathrm{Penalty}}(u_h,\, p_h\,;\, v_h,\, q_h) &:= a(u_h, v_h) + b(v_h, p_h) + b(u_h, q_h) - \delta d(p_h, q_h)\,, \end{aligned}$$

where each δ is a positive constant. Since u_h and v_h belong to the P1 finite element space, $\nabla \cdot D(u_h)$ and $\nabla \cdot D(v_h)$ vanish. We have

$$A_\delta^{\mathrm{Penalty}}(u_h,\, p_h\,;\, v_h,\, q_h) = A_\delta^{\mathrm{GLS}}(u_h,\, p_h\,;\, v_h,\, q_h)\,,$$

for all $(u_h, p_h), (v_h, q_h) \in X_h \times S_h$. Therefore, we write the bilinear form without the symbols "Penalty" or "GLS" as $A_\delta(u_h,\, p_h\,;\, v_h,\, q_h)$.

In the *Dirichlet Problem* we define functionals F_δ^{GLS} and $F_\delta^{\mathrm{Penalty}}$ on $X_h \times S_h$ by

$$F_\delta^{\mathrm{GLS}}(v_h,\, q_h) := (f, v_h) - \delta \sum_{K \in \mathcal{T}_h} h_K^2 (f, \nabla q_h\,;\, K)\,, \tag{1.38}$$

$$F_\delta^{\mathrm{Penalty}}(v_h,\, q_h) := (f, v_h)\,. \tag{1.39}$$

A stabilized finite element problem of the GLS type for (1.1) and (1.2) is to find $(u_h, p_h) \in V_h(g) \times Q_h$ satisfying

$$A_\delta(u_h,\, p_h\,;\, v_h,\, q_h) = F_\delta^{\mathrm{GLS}}(v_h,\, q_h) \tag{1.40}$$

for all $(v_h, q_h) \in V_h \times Q_h$, and that of the penalty type for (1.1) and (1.2) is to find $(u_h, p_h) \in V_h(g) \times Q_h$ satisfying

$$A_\delta(u_h,\, p_h\,;\, v_h,\, q_h) = F_\delta^{\mathrm{Penalty}}(v_h,\, q_h) \tag{1.41}$$

for all $(v_h, q_h) \in V_h \times Q_h$.

In the *Slip Problem* we consider the same finite element approximations as (1.40) and (1.41) to (1.1) and (1.3). Here we use bilinear forms $a_h(\cdot,\cdot)$, $b_h(\cdot,\cdot)$, $d_h(\cdot,\cdot)$, the inner product $(\cdot,\cdot)_h$ on the domain approximated by the polyhedron: Ω_h, and functionals $F_\delta^{\rm GLS}$ and $F_\delta^{\rm Penalty}$ defined by

$$F_\delta^{\rm GLS}(v_h,\, q_h) := (f, v_h)_h + 2\int_{\Gamma_h} g_t \cdot v_h \, ds - \delta \sum_{K\in\mathcal{T}_h} h_K^2(f, \nabla q_h \,;\, K)\,,$$

$$F_\delta^{\rm Penalty}(v_h,\, q_h) := (f, v_h)_h + 2\int_{\Gamma_h} g_t \cdot v_h \, ds\,.$$

A solution should be found in $\tilde{V}_h(g_n) \times \tilde{Q}_h$.

Error estimates of finite element solutions

Following Franca–Stenberg [22], we will review error estimates for the GLS method and the penalty method of the Stokes problems.

Lemma 1.10
(i) *$a(\cdot,\cdot)$ is uniformly coercive i.e., there exists a positive constant α_0 such that for all positive number h*

$$\inf_{v_h\in V_h} \frac{a(v_h, v_h)}{||v_h||_1^2} \geq \alpha_0\,.$$

(ii) *$b(\cdot,\cdot)$ satisfies the uniform weak inf-sup condition, i.e., there exist positive constants β_0 and β_1 such that for all positive number h and $q_h \in Q_h$*

$$\sup_{v_h\in V_h} \frac{b(v_h, q_h)}{||v_h||_1} \geq \beta_0||q_h||_0 - \beta_1|q_h|_h\,.$$

Since $V_h \subset V$, (i) is trivial. A proof of the uniform weak inf-sup condition (ii) is found in [22].

In the *Slip Problem* almost same results on the coerciveness and the uniform weak inf-sup condition hold on function spaces $\tilde{V}_h$ defined in (1.34) and $\tilde{Q}_h$ defined in (1.35).

Lemma 1.11
(i) *There exist positive constants α_0 and h_0 such that for all $h \in (0, h_0]$*

$$\inf_{v_h\in \tilde{V}_h} \frac{a_h(v_h, v_h)}{||v_h||_1^2} \geq \alpha_0\,.$$

(ii) *There exist positive constants β_0, β_1, and h_0 such that for all $h \in (0, h_0]$ and $q_h \in \tilde{Q}_h$*

$$\sup_{v_h\in \tilde{V}_h} \frac{b_h(v_h, q_h)}{||v_h||_1} \geq \beta_0||q_h||_0 - \beta_1|q_h|_h\,.$$

A proof of the uniform coerciveness of $a_h(\cdot,\cdot)$ is found in [44] and a proof of the uniform weak inf-sup condition of $b_h(\cdot,\cdot)$ is found in [46].

Proposition 1.2 *We have*

$$A_\delta(v_h,\, q_h\,;\, v_h,\, -q_h) \geq \alpha_0 ||v_h||_1^2 + \delta\, |q_h|_h^2\,, \tag{1.42}$$

for all $v_h \in V_h$ *and* $q_h \in Q_h$.

Lemma 1.12
(i) $A_\delta(\cdot,\, \cdot\,;\, \cdot,\, \cdot)$ *is uniformly bounded, i.e., there exists a positive constant* A_1 *such that for all positive* h *and* $(u_h, p_h), (v_h, q_h) \in V_h \times Q_h$ *that*

$$A_\delta(u_h,\, p_h\,;\, v_h,\, q_h) \leq A_1 ||(u_h, p_h)||_{V\times Q} ||(v_h, q_h)||_{V\times Q}\,.$$

(ii) $A_\delta(\cdot,\, \cdot\,;\, \cdot,\, \cdot)$ *satisfies a stability inequality, i.e., there exists a positive constant* α_1 *such that for all positive* h

$$\inf_{(u_h,p_h)\in V_h\times Q_h} \sup_{(v_h,q_h)\in V_h\times Q_h} \frac{A_\delta(u_h,\, p_h\,;\, v_h,\, q_h)}{||(u_h, p_h)||_{V\times Q} ||(v_h, q_h)||_{V\times Q}} \geq \alpha_1\,.$$

Proof. The uniform boundedness (i) is easily obtained from the Schwarz inequality and (1.36). The stability inequality (ii) follows from the uniform coerciveness of $a(\cdot,\cdot)$, the uniform weak inf-sup condition of $b(\cdot,\cdot)$, the inequality (1.42), and the uniform boundedness (i). A complete proof is found in [22]. □

Theorem 1.4
(i) *There exists a unique solution* $(u_h, p_h) \in V_h(g) \times Q_h$ *of the problem* (1.40) *or* (1.41) *for all positive* δ.
(ii) *Let* $(u, p) \in H^2(\Omega)^3 \times H^1(\Omega)$ *be the solution of* (1.1) *and* (1.2). *Let* $(u_h, p_h) \in V_h(g) \times Q_h$ *be the solution of the problem* (1.40) *or* (1.41). *There exists a positive constant* c *such that for all positive* h

$$||u - u_h||_1 + ||p - p_h||_0 \leq c\, h\{||u||_2 + ||p||_1\}\,.$$

Proof. Existence of a unique solution of problem (1.40) or (1.41) follows from Lemma 1.12 (ii) and Lemma 1.4.

In case of the GLS type, let $(u_h, p_h) \in V_h(g) \times Q_h$ be the solution of (1.40). Let $v_h \in V_h(g)$ and $q_h \in Q_h$ be arbitrary functions. We have a consistency property on

the bilinear form $A_\delta^{\mathrm{GLS}}(\cdot,\cdot)$ as follows:

$$\begin{aligned}
A_\delta^{\mathrm{GLS}}&(u-u_h,\, p-p_h\,;\, v_h,\, q_h)\\
&= a(u,v_h)+b(v_h,p)+b(u,q_h)\\
&\quad -\delta\sum_{K\in\mathcal{T}_h} h_K^2(2\nabla\cdot D(u)+\nabla p, 2\nabla\cdot D(v_h)+\nabla q_h\,;\,K)\\
&\quad -\Big\{\, a(u_h,v_h)+b(v_h,p_h)+b(u_h,q_h)\\
&\quad -\delta\sum_{K\in\mathcal{T}_h} h_K^2(2\nabla\cdot D(u_h)+\nabla p_h, 2\nabla\cdot D(v_h)+\nabla q_h\,;\,K)\,\Big\}\\
&= (f,v_h)-\delta\sum_{K\in\mathcal{T}_h} h_K^2(2\nabla\cdot D(u)+\nabla p, 2\nabla\cdot D(v_h)+\nabla q_h\,;\,K)\\
&\quad -\Big\{\,(f,v_h)-\delta\sum_{K\in\mathcal{T}_h} h_K^2(f, 2\nabla\cdot D(v_h)+\nabla q_h\,;\,K)\,\Big\}\\
&= -\delta\sum_{K\in\mathcal{T}_h} h_K^2(2\nabla\cdot D(u)+\nabla p-f, 2\nabla\cdot D(v_h)+\nabla q_h\,;\,K)\\
&= 0\,.
\end{aligned}$$

Here the last equality is obtained from the fact that $(u,p)\in H^2(\Omega)^3\times H^1(\Omega)$ is the solution of (1.1) and (1.2). Using the Schwarz inequality on the inner product on each element K, we obtain

$$\begin{aligned}
\sup_{(w_h,r_h)\in V_h\times Q_h}&\frac{A_\delta^{\mathrm{GLS}}(u-v_h,\, p-q_h\,;\, w_h,\, r_h)}{||(w_h,r_h)||_{V\times Q}}\\
&\le A_2\left(||(u-v_h,p-q_h)||_{V\times Q}+h||\nabla\cdot D(u)||_0+|p-q_h|_h\right)\,, \quad (1.43)
\end{aligned}$$

where A_2 is dependent on δ, but independent of h. Combining Lemma 1.12 (ii) and the inequality (1.43), we obtain the following estimate from Theorem 1.2 (ii).

$$||(u-u_h,p-p_h)||_{V\times Q}\le\left(1+\frac{A_2}{\alpha_1}\right)||(u-v_h,p-q_h)||_{V\times Q}+\frac{A_2}{\alpha_1}\left(\,h||\nabla\cdot D(u)||_0+|p-q_h|_h\,\right)\,.$$

The assertion follows from Lemma 1.8 and Lemma 1.9 (ii).

In case of the penalty type, let $(u_h,p_h)\in V_h(g)\times Q_h$ be the solution of (1.41). Let $v_h\in V_h(g)$ and $q_h\in Q_h$ be arbitrary functions. Then we have

$$A_\delta(u-u_h,\, p-p_h\,;\, v_h,\, q_h)=-\delta d(p,q_h)\,.$$

Using the Schwarz inequality on the inner product on each element K, we obtain

$$\begin{aligned}
|A_\delta(u-u_h,\, p-p_h\,;\, v_h,\, q_h)| &\le \delta|d(p,q_h)|\\
&\le \delta\{\sum_{K\in\mathcal{T}_h} h_K^2(\nabla p,\nabla p\,;\,K)\}^{1/2}|q_h|_h\\
&\le c_0 h||\nabla p||_0||q_h||_0\,,
\end{aligned}$$

where $c_0 > 0$. Here the last inequality follows from the inequality (1.36). We have

$$\sup_{(w_h,r_h)\in V_h\times Q_h} \frac{A_\delta(u-v_h,\, p-q_h\,;\, w_h,\, r_h)}{||(w_h,r_h)||_{V\times Q}} \le A_3\left(||(u-v_h,p-q_h)||_{V\times Q} + |p-q_h|_h\right) ,$$

where A_3 is dependent on δ, but independent of h. From Theorem 1.2 (i), we get

$$||(u-u_h,p-p_h)||_{V\times Q} \le \left(1+\frac{A_3}{\alpha_1}\right)||(u-v_h,p-q_h)||_{V\times Q} + \frac{A_3}{\alpha_1}|p-q_h|_h + h\,\frac{c_0}{\alpha_1}||\nabla p||_0 .$$

This estimate leads to the desired result. □

In the *Slip Problem*, a similar result as Theorem 1.4 holds.

Theorem 1.5
(i) *There exists a positive constant* h_0 *such that for all* $h \in (0, h_0]$, *problem* (1.40) *or* (1.41) *has a unique solution* $(u_h, p_h) \in \tilde{V}_h(g_n) \times \tilde{Q}_h$.
(ii) *Let* $(u, p) \in H^2(\Omega)^3 \times H^1(\Omega)$ *be the solution of* (1.1) *and* (1.3). *There exist positive constants* c *and* h_0 *such that for all* $h \in (0, h_0]$

$$||u-u_h||_1 + ||p-p_h||_0 \le c\,h\{||u||_2 + ||p||_1\} .$$

Remark 1.5 We note that the weak form of the Stokes equations is of a saddle-point type, while the discretized variational problem with stabilization terms is not. The stability inequality in Lemma 1.12 (ii) plays a key role to guarantee the existence and uniqueness of the solution of the discretized variational problem.

1.6 Summary

In this chapter we have reviewed stabilized finite element methods for the Stokes equations with Dirichlet boundary conditions or slip boundary conditions. Considering the cost of 3-D computation, we used P1/P1 finite element approximation to the Stokes problem. Because P1/P1 combination does not satisfy the inf-sup condition, a stabilization technique of the GLS type or the penalty type was used. We have presented an abstract formulation of the finite element method including stabilization techniques, and have shown a finite element equation by discretized variational form in the solution space with stiffness matrix on the whole nodes. Unfortunately, the conventional solver in finite element codes has application limitations to finite element equations and is not applicable to general problems. Therefore, we use a Krylov subspace method that is applicable to any cases of finite element equations and is suitable for large-scale 3-D problems. We will show details of the solver for the Stokes problems in the next chapter.

Chapter 2

Parallel Solver for the Stokes Equations by Domain Decomposition and Iterative Methods

2.1 Introduction

Because it is required to produce a finite element solution of a large-scale 3-D problem in reasonable computation time with current computer resources, the solver should perform fast computation with small memory storage. Therefore, it is very important to construct an efficient solver for the finite element equations. It is also necessary that its algorithm should be simple for ease of implementation, and should be applicable to various problems. In the Stokes problem, several types of boundary conditions appear. An answer for fast computation is to use the conjugate gradient (CG) method [26], which has low computation cost and can be parallelized easily. We add projection operations to the CG method to treat boundary conditions and linear constraints. The CG method with a projection also satisfies the requirement for ease of implementation. To reduce memory requirements, we use a domain decomposition into congruent subdomains.

Conjugate gradient method with orthogonal projection

Using mixed finite element methods, the matrix form of the Stokes problem is written as

$$\begin{pmatrix} A & B^T \\ B & 0 \end{pmatrix} \begin{pmatrix} \vec{u} \\ \vec{p} \end{pmatrix} = \begin{pmatrix} \vec{f} \\ 0 \end{pmatrix} ,$$

where stiffness matrices A and B correspond to bilinear forms $a(\cdot,\cdot)$ and $b(\cdot,\cdot)$, respectively, and a load vector $\vec{f}$ corresponds to an external force. By elimination

of velocity, we obtain a linear system of pressure

$$S\vec{p} = \vec{g}\,,$$

where S is a Schur complement defined by $BA^{-1}B^T$ and $\vec{g} = BA^{-1}\vec{f}$. When the kernel of the transposed matrix B^T is an empty set, S is positive definite and the CG method is applicable to the linear system. An efficient preconditioner is proposed in [16]. This method is called the pressure-matrix method [36]. We implemented this method on a parallel computer for 2-D problems [41]. The pressure-matrix method is also applicable to finite element equations obtained using stabilization techniques. However, there are two difficulties with this method. One is how to construct the matrix A that is positive definite and the matrix B whose transposed matrix B^T has the zero-dimension kernel with low computational cost for various types of boundary conditions. Unfortunately, in the problem considered in a spherical shell domain with the slip boundary conditions imposed, the kernel of the coefficient matrix of the Stokes problem has a dimension of four, corresponding to rigid body rotations and pressure lifting. The other difficulty is that, in a solver that consists of nested loops, an outer loop for pressure and an inner loop for velocity, execution of the nested loop may be slower than a single loop combining velocity and pressure.

We present a solver for the Stokes problems in the framework described in Section 1.3.2, where a discretized variational equation in the solution space is considered. Our solver deals with the discretized equations expressed with a stiffness matrix that is defined on all finite element nodes independently of the boundary conditions, and an orthogonal projection onto the solution space. In the solution space for the Stokes equations in a spherical shell domain with slip boundary conditions imposed, normal components of the velocity on the boundary nodes vanish and the rigid body rotations and pressure lifting are eliminated. Unfortunately, the generalized solver of conventional finite element codes, which is described precisely in Appendix A.1, is not applicable to the Stokes problem with slip boundary conditions. The solver can be applied under the condition that the dimension of the image of the matrix should be equal to the dimension of the solution space. However, this condition is not satisfied in the case of the Stokes problem with slip boundary conditions, because of perturbation from approximations by finite elements. On the other hand, our solver can deal with different types of boundary conditions in the same framework, and can be applied to the Stokes problem with slip boundary conditions.

Because the coefficient matrix of the Stokes problem with both unknown velocity and pressure is symmetric, although it is indefinite, we use the CG method. We also employ a preconditioning technique to improve the convergence of the CG method. Here an indefinite matrix is used as a preconditioning matrix. It is known that some algorithms can solve linear systems with indefinite symmetric matrices. The SYMMLQ and MINRES methods have been developed [34]; in these algorithms some preconditioners with positive definite matrices are applicable. For linear systems with unsymmetric matrices, the CSBCG method [4] has been proposed as a variant of the bi-conjugate gradient (BiCG) method [20] to avoid a kind of 'break-

down' in the execution of the method, and it is applicable to symmetric matrices. Such breakdowns of the BiCG method were investigated by Saad [39] and Joubert [31]. The CG method consists of the simplest procedure in the Krylov subspace methods, however, it may stop without finding any solution of a linear equation with an indefinite matrix. Following works of Saad and Joubert, we obtain a necessary and sufficient condition for no-breakdown of the preconditioned CG (PCG) method. We verify that when a breakdown does not occur, the PCG method can find a solution in a Krylov subspace.

A domain decomposition into congruent subdomains

For domains having some form of symmetry or periodicity, several computational techniques have been developed to reduce the memory requirements. When a domain has a form of periodicity, by assuming a periodic solution, Zienkiewicz [51] reduced the problem to one in a unit subdomain. When a domain has a form of symmetry, Bossavit [8] presented a strategy to reduce the size of a finite element problem by the group representation theory. Bonnet [9] extended this to the boundary integral equation method. In their approaches, no symmetry is assumed for the solution, although there is a restriction in the domain decomposition. Every subproblem should have the same type of boundary conditions imposed. We assume that a domain is decomposed into congruent subdomains that are images of a reference domain transformed by orthogonal matrices. Decompositions of the finite element matrix are performed independently of the boundary conditions. Therefore, more flexible decompositions can be adopted. Because the boundary conditions are treated by the projection operation, this strategy of the decomposition can be adopted. Unfortunately, multiplications of the orthogonal matrix at each nodal point, which cause additional costs, are necessary to transform stiffness matrices for velocity unknowns. If a domain has a form of symmetry, it can be decomposed into congruent subdomains that are transformed by orthogonal matrices whose components consist of -1, 0, and 1. Using this domain decomposition, the finite element matrices in the transformed subdomains are expressed by a sub-matrix in the reference subdomain by renumbering indices and changing signs. By this property, a finite element matrix in the whole domain can be produced from a sub-matrix in the reference subdomain. Therefore, we can drastically reduce the memory requirements using the domain decomposition into congruent subdomains.

Organization of this chapter

In Section 2.2, we present the finite element equation with an orthogonal projection for the Stokes problem with Dirichlet boundary conditions or slip boundary conditions. Second, we show the construction of orthogonal projections for both boundary conditions. In Section 2.3, we investigate the preconditioned conjugate gradient (PCG) method for indefinite symmetric matrices. We will show that with successful execution of the PCG method, a solution is obtained in a Krylov subspace.

Then, we introduce a preconditioner to the finite element equations of the Stokes problems and demonstrate the convergence of the PCG method by numerical experiments. In Section 2.4, we show an algorithm to reduce the memory requirements for the stiffness matrices using domain decomposition into congruent subdomains. The properties of the transformation of the scalar- and vector-valued finite element bases and the finite element matrices are presented. We show numerical results of reducing the memory for stiffness matrix in a spherical shell domain. In Section 2.5, we present a parallel computation algorithm for a multiplication of a matrix by a vector using domain decomposition into congruent subdomains. We show that the domain decomposition algorithm is useful not only in parallel computation with a shared-memory parallel computer, but also in serial computations with workstations and vector computers.

2.2 Finite element equation with orthogonal projection that deals with boundary conditions and linear constraints

2.2.1 Finite element basis and finite element matrices

Finite element basis

Let n_G be the number of nodes in $\bar{\Omega}$. We define an index set $\Lambda_G := \{1, 2, \ldots, n_G\}$ and denote a node by P_μ for $\mu \in \Lambda_G$. Let $\Lambda_\Gamma \subset \Lambda_G$ be an index set of node on the boundary Γ :

$$P_\mu \in \Gamma \quad (\mu \in \Lambda_\Gamma)\,.$$

We set $n_\Gamma := \#\Lambda_\Gamma$, where $\#$ stands for the total number of elements of the set. Let Λ_S be an index set defined by $\{1, 2, \ldots, n_S\}$, where $n_S = n_G$ and $\{\psi_\mu\}_{\mu\in\Lambda_S}$ be a finite element basis of S_h (scalar-valued P1 finite element space, see Section 1.5 for its definition). The finite element basis satisfies

$$\psi_\mu(P_\nu) = \delta_{\mu\,\nu} \qquad (\mu,\, \nu \in \Lambda_G)\,, \tag{2.1}$$

where $\delta_{\mu\,\nu}$ is the Kronecker delta. Let $\Lambda_X := \{1, 2, \ldots, n_X\}$ be an index set and $\{\varphi_\alpha\}_{\alpha\in\Lambda_X}$ be a finite element basis of X_h (vector-valued P1 finite element space). We associate a pair $[\alpha_0,\, k]$ of a node index and a component index with an index $\alpha \in \Lambda_X$ and identify them as

$$\alpha = [\alpha_0,\, k] \qquad (\alpha_0 \in \Lambda_G,\ k \in \{1, 2, 3\})\,.$$

The basis of X_h satisfies

$$[\varphi_\alpha(P_{\beta_0})]_l = \delta_{\alpha\,\beta} \qquad (\alpha,\, \beta = [\beta_0,\, l] \in \Lambda_X)\,.$$

We note that the vector-valued basis is expressed by the scalar-valued basis as

$$[\varphi_\alpha]_l = \delta_{k\,l}\psi_{\alpha_0} \quad (\alpha = [\alpha_0,\, k] \in \Lambda_X, l = 1, 2, 3)\,. \tag{2.2}$$

Finite element matrices

Let A, B, D, M_1 and M_0 be stiffness and mass matrices defined by

$$[A]_{\alpha\,\beta} := a(\varphi_\beta, \varphi_\alpha\,;\,\Omega) \qquad (\alpha, \beta \in \Lambda_X)\,, \tag{2.3a}$$
$$[B]_{\mu\,\beta} := b(\varphi_\beta, \psi_\mu\,;\,\Omega) \qquad (\mu \in \Lambda_S,\ \beta \in \Lambda_X)\,, \tag{2.3b}$$
$$[M_1]_{\alpha\,\beta} := (\varphi_\beta, \varphi_\alpha\,;\,\Omega) \qquad (\alpha, \beta \in \Lambda_X)\,, \tag{2.3c}$$
$$[D]_{\mu\,\nu} := d(\psi_\nu, \psi_\mu\,;\,\Omega) \qquad (\mu, \nu \in \Lambda_S)\,, \tag{2.3d}$$
$$[M_0]_{\mu\,\nu} := (\psi_\nu, \psi_\mu\,;\,\Omega) \qquad (\mu, \nu \in \Lambda_S)\,. \tag{2.3e}$$

We note that the sizes of matrices A, B, M_1, D, and M_0 are $n_X \times n_X$, $n_S \times n_X$, $n_X \times n_X$, $n_S \times n_S$, and $n_S \times n_S$, respectively. In the *Slip Problem*, the stiffness and mass matrices are computed in Ω_h. The stiffness and mass matrices are independent of the boundary conditions, therefore each matrix is generated by the same procedure in both the *Dirichlet Problem* and *Slip Problem* except for the difference of the domains concerned, Ω and Ω_h.

2.2.2 Solution space for a finite element equation

Case of Dirichlet boundary conditions

In the *Dirichlet Problem* we define an index set Λ_D for the Dirichlet data on the boundary by

$$\Lambda_D := \{\beta = [\beta_0,\, \beta_1] \in \Lambda_X\ ;\ \beta_0 \in \Lambda_\Gamma,\ \beta_1 \in \{1, 2, 3\}\}\,.$$

We put $n_D := \#\Lambda_D = 3n_\Gamma$. We define an affine space $\vec{V}(g)$, and vector spaces $\vec{V}$ and $\vec{Q}$ by

$$\begin{aligned}
\vec{V}(g) &:= \{\vec{v} \in \mathbb{R}^{n_X}\,;\, (\vec{v},\, \vec{e}^{\,(\beta)})_{\mathbb{R}^{n_X}} = [g(P_{\beta_0})]_{\beta_1} \quad (\,\beta \in \Lambda_D\,)\,\}\,,\\
\vec{V} &:= \vec{V}(0)\,,\\
\vec{Q} &:= \{\vec{q} \in \mathbb{R}^{n_S}\,;\, (\vec{q},\, M_0\vec{1}\,)_{\mathbb{R}^{n_S}} = 0\}\,,
\end{aligned}$$

corresponding to the finite element affine space $V_h(g)$ and the finite element spaces V_h and Q_h, respectively. Dimensions of $\vec{V}$ and $\vec{Q}$ are denoted by n_V and n_Q, respectively. We note that $n_V = n_X - n_D$ and $n_Q = n_S - 1$. Here $\vec{e}^{\,(\beta)} \in \mathbb{R}^{n_X}$ is a unit vector: $[\vec{e}^{\,(\beta)}]_\gamma = \delta_{\beta\,\gamma}$, and $\vec{1} = (1, 1, \ldots, 1)^T$. Let $P_{\vec{V}}$ and $P_{\vec{Q}}$ be the orthogonal projections defined by

$$\begin{aligned}
(P_{\vec{V}}\vec{u},\, \vec{v}\,)_{\mathbb{R}^{n_X}} &= (\vec{u},\, \vec{v}\,)_{\mathbb{R}^{n_X}} \quad \text{for all } \vec{v} \in \vec{V}\,,\\
(P_{\vec{Q}}\vec{p},\, \vec{q}\,)_{\mathbb{R}^{n_S}} &= (\vec{p},\, \vec{q}\,)_{\mathbb{R}^{n_S}} \quad \text{for all } \vec{q} \in \vec{Q}\,.
\end{aligned}$$

The projections are expressed as

$$\begin{aligned}
P_{\vec{V}}\vec{u} &= \vec{u} - \sum_{\beta \in \Lambda_D} (\vec{u},\, \vec{e}^{\,(\beta)})_{\mathbb{R}^{n_X}}\vec{e}^{\,(\beta)}\,,\\
P_{\vec{Q}}\vec{p} &= \vec{p} - (\vec{p},\, \vec{m})_{\mathbb{R}^{n_S}}\vec{m}\,,
\end{aligned}$$

where $\vec{m} := M_0\vec{1}/||M_0\vec{1}||_{\mathbb{R}^{n_S}}$. For the GLS type method, we define load vectors $\vec{f_1} \in \mathbb{R}^{n_X}$ and $\vec{f_2} \in \mathbb{R}^{n_S}$ by

$$\begin{aligned} [\vec{f_1}]_\alpha &:= (f, \varphi_\alpha) && (\alpha \in \Lambda_X)\,, \\ [\vec{f_2}]_\mu &:= -\delta \sum_{K \in \mathcal{T}_h} h_K^2 (f, \nabla\psi_\mu\,;\, K) && (\mu \in \Lambda_S)\,. \end{aligned}$$

For the penalty type method we set $\vec{f_2} = 0$.

Case of slip boundary conditions

In the *Slip Problem* we prepare vectors $\{\vec{n}^{(j)}\}_{j\in\Lambda_\Gamma} \subset \mathbb{R}^{n_X}$ corresponding to the unit outer normal $n_\Omega \in \mathbb{R}^3$ to the boundary and $\{\vec{v}^{(j)}\}_{j=1,2,3} \subset \mathbb{R}^{n_X}$ corresponding to the rigid body rotations defined by

$$[\vec{n}^{(j)}]_\alpha := \delta_{j\,\alpha_0}[n_\Omega(P_j)]_{\alpha_1} \qquad (j \in \Lambda_\Gamma\,,\ \alpha = [\alpha_0, \alpha_1] \in \Lambda_X)\,, \tag{2.4}$$

$$[\vec{v}^{(j)}]_\alpha := [v^{(j)}(P_{\alpha_0})]_{\alpha_1} \qquad (j \in \{1,2,3\}\,, \alpha = [\alpha_0, \alpha_1] \in \Lambda_X)\,. \tag{2.5}$$

We note that

$$(\vec{n}^{(i)},\, \vec{n}^{(j)})_{\mathbb{R}^{n_X}} = \delta_{i\,j} \quad (i, j \in \Lambda_\Gamma).$$

We define affine spaces $\vec{W}(g_n)$, $\vec{V}(g_n)$ and vector spaces $\vec{W}$, $\vec{V}$, and $\vec{Q}$ by

$$\begin{aligned} \vec{W}(g_n) &:= \{\vec{v} \in \mathbb{R}^{n_X}\ ;\ (\vec{v},\, \vec{n}^{(j)})_{\mathbb{R}^{n_X}} = g_n(P_j) \quad (j \in \Lambda_\Gamma)\}\,, \\ \vec{W} &:= \vec{W}(0)\,, \\ \vec{V}(g_n) &:= \{\vec{v} \in \vec{W}(g_n)\ ;\ (M_1\vec{v},\, \vec{v}^{(j)})_{\mathbb{R}^{n_X}} = 0 \quad (j \in \{1,2,3\})\}\,, \\ \vec{V} &:= \vec{V}(0)\,, \\ \vec{Q} &:= \{\vec{q} \in \mathbb{R}^{n_S}\ ;\ (M_0\vec{q},\, \vec{1})_{\mathbb{R}^{n_S}} = 0\}\,. \end{aligned}$$

Dimensions of $\vec{V}$ and $\vec{Q}$ are denoted by n_V and n_Q, respectively. We note that $n_V = n_X - n_\Gamma - 3$ and $n_Q = n_S - 1$. Let $P_{\vec{V}}$ and $P_{\vec{Q}}$ be the orthogonal projections defined by

$$\begin{aligned} (P_{\vec{V}}\vec{u},\, \vec{v}\,)_{\mathbb{R}^{n_X}} &= (\vec{u},\, \vec{v}\,)_{\mathbb{R}^{n_X}} \quad \text{for all } \vec{v} \in \vec{V}\,, \\ (P_{\vec{Q}}\vec{p},\, \vec{q}\,)_{\mathbb{R}^{n_S}} &= (\vec{p},\, \vec{q}\,)_{\mathbb{R}^{n_S}} \quad \text{for all } \vec{q} \in \vec{Q}\,. \end{aligned}$$

Let vectors $\{\vec{w}^{(j)}\}_{j=1,2,3}$ satisfy

$$\begin{aligned} \mathrm{span}[\{\vec{n}^{(i)}\}_{i\in\Lambda_\Gamma},\, \{\vec{w}^{(j)}\}_{j=1,2,3}] &= \mathrm{span}[\{\vec{n}^{(i)}\}_{i\in\Lambda_\Gamma},\, \{M_1\vec{v}^{(j)}\}_{j=1,2,3}]\,, \\ (\vec{w}^{(j)},\, \vec{n}^{(i)})_{\mathbb{R}^{n_X}} &= 0 \quad (j = 1,2,3,\ i \in \Lambda_\Gamma)\,, \\ (\vec{w}^{(j)},\, \vec{w}^{(i)})_{\mathbb{R}^{n_X}} &= \delta_{i\,j} \quad (i, j = 1,2,3)\,, \end{aligned}$$

and let $\vec{m} := M_0\vec{1}/||M_0\vec{1}||_{\mathbb{R}^{n_S}}$. Here $\{\vec{w}^{(j)}\}_{j=1,2,3}$ can be generated by the Gram–Schmidt orthogonization process. Then the projections are expressed as

$$P_{\vec{V}}\vec{u} = \vec{u} - \sum_{i\in\Lambda_\Gamma} (\vec{u},\, \vec{n}^{(i)})_{\mathbb{R}^{n_X}}\vec{n}^{(i)} - \sum_{1\le j\le 3} (\vec{u},\, \vec{w}^{(j)})_{\mathbb{R}^{n_X}}\vec{w}^{(j)}\,,$$
$$P_{\vec{Q}}\vec{p} = \vec{p} - (\vec{p},\, \vec{m})_{\mathbb{R}^{n_S}}\vec{m}\,.$$

We define load vectors $\vec{f_1} \in \mathbb{R}^{n_X}$ and $\vec{f_2} \in \mathbb{R}^{n_S}$ for the GLS type method by

$$[\vec{f_1}]_\alpha := (f, \varphi_\alpha)_h + 2\int_{\Gamma_h} g_t \cdot \varphi_\alpha\, ds \qquad (\alpha \in \Lambda_X)\,,$$
$$[\vec{f_2}]_\mu := -\delta \sum_{K\in\mathcal{T}_h} h_K^2(f, \nabla\psi_\mu\,;\, K) \qquad (\mu \in \Lambda_S)\,.$$

For the penalty type method we set $\vec{f_2} = 0$.

2.2.3 A Finite element equation of the Stokes problem

The stabilized finite element method (1.40) or (1.41) is equivalent to a problem to find $(\vec{u}, \vec{p}) \in \vec{V}(g) \times \vec{Q}$ satisfying

$$(A\vec{u},\, \vec{v})_{\mathbb{R}^{n_X}} + (B\vec{v},\, \vec{p})_{\mathbb{R}^{n_S}} = (\vec{f_1},\, \vec{v})_{\mathbb{R}^{n_X}}\,, \tag{2.6a}$$
$$(B\vec{u},\, \vec{q})_{\mathbb{R}^{n_S}} - \delta(D\vec{p},\, \vec{q})_{\mathbb{R}^{n_S}} = (\vec{f_2},\, \vec{q})_{\mathbb{R}^{n_S}}\,, \tag{2.6b}$$

for all $(\vec{v}, \vec{q}) \in \vec{V} \times \vec{Q}$. In the penalty method $\vec{f_2} = 0$. This pair of the equations can be rewritten in a matrix form including the orthogonal projections, as to find $(\vec{u}_0, \vec{p}) \in \vec{V} \times \vec{Q}$ satisfying

$$\begin{pmatrix} P_{\vec{V}} & 0 \\ 0 & P_{\vec{Q}} \end{pmatrix} \begin{pmatrix} A & B^T \\ B & -\delta D \end{pmatrix} \begin{pmatrix} P_{\vec{V}} & 0 \\ 0 & P_{\vec{Q}} \end{pmatrix} \begin{pmatrix} \vec{u}_0 \\ \vec{p} \end{pmatrix} = \begin{pmatrix} P_{\vec{V}}(\vec{f_1} - A\vec{u}_g) \\ P_{\vec{Q}}(\vec{f_2} - B\vec{u}_g) \end{pmatrix}\,, \tag{2.7}$$

where $\vec{u}_g$ belongs to $\vec{V}(g)$. Then, $(\vec{u}, \vec{p}) = (\vec{u}_0 + \vec{u}_g, \vec{p})$ is a solution of the problem (2.6).

We define an $(n_X + n_S) \times (n_X + n_S)$ matrix $\mathcal{A}$ to represent the coefficient matrix in the linear system (2.7),

$$\mathcal{A} := \begin{pmatrix} P_{\vec{V}} & 0 \\ 0 & P_{\vec{Q}} \end{pmatrix} \begin{pmatrix} A & B^T \\ B & -\delta D \end{pmatrix} \begin{pmatrix} P_{\vec{V}} & 0 \\ 0 & P_{\vec{Q}} \end{pmatrix}\,.$$

We can find numbers of positive, zero and negative eigenvalues of $\mathcal{A}$. Let $\{\lambda_i\}$ be eigenvalues of $\mathcal{A}$.

Proposition 2.1
(i) $\mathcal{A}$ *is regular in* $\vec{V} \times \vec{Q}$.
(ii) *It holds that*

$$\begin{aligned}
&\#\{\lambda_i > 0\} = n_V \,, \\
&\#\{\lambda_i = 0\} = (n_X + n_S) - (n_V + n_Q) \,, \\
&\#\{\lambda_i < 0\} = n_Q \,.
\end{aligned}$$

Here $n_V = n_X - n_D$ *in the Dirichlet Problem,* $n_V = n_X - n_\Gamma - 3$ *in the Slip Problem, and* $n_Q = n_S - 1$ *in both cases.*
Proof. The statement (i) follows from the property of the linear system of finite element equations in the abstract framework described in Section 1.3.4. We will prove only (ii). Since A is regular in $\vec{V}$ and D is regular in $\vec{Q}$, we have

$$\begin{aligned}
(P_{\vec{V}} A P_{\vec{V}})^\dagger (P_{\vec{V}} A P_{\vec{V}}) = P_{\vec{V}} \,, \\
(P_{\vec{Q}} D P_{\vec{Q}})^\dagger (P_{\vec{Q}} D P_{\vec{Q}}) = P_{\vec{Q}} \,.
\end{aligned}$$

Therefore we obtain

$$\begin{aligned}
\mathcal{A} &= \begin{pmatrix} P_{\vec{V}} A P_{\vec{V}} & P_{\vec{V}} B^T P_{\vec{Q}} \\ P_{\vec{Q}} B P_{\vec{V}} & -\delta P_{\vec{Q}} D P_{\vec{Q}} \end{pmatrix} \\
&= \begin{pmatrix} I_{n_X} & 0 \\ (P_{\vec{Q}} B P_{\vec{V}})(P_{\vec{V}} A P_{\vec{V}})^\dagger & I_{n_S} \end{pmatrix} \begin{pmatrix} P_{\vec{V}} A P_{\vec{V}} & 0 \\ 0 & -S_D \end{pmatrix} \begin{pmatrix} I_{n_X} & (P_{\vec{V}} A P_{\vec{V}})^\dagger (P_{\vec{V}} B^T P_{\vec{Q}}) \\ 0 & I_{n_S} \end{pmatrix} ,
\end{aligned}$$

where I_n is the identity matrix in $\mathbb{R}^n$ and S_D is a Schur complement matrix of $\mathcal{A}$ defined as

$$S_D := \delta P_{\vec{Q}} D P_{\vec{Q}} + (P_{\vec{Q}} B P_{\vec{V}})(P_{\vec{V}} A P_{\vec{V}})^\dagger (P_{\vec{V}} B^T P_{\vec{Q}}) \,.$$

Since A is positive definite on $\vec{V}$ and D is positive definite on $\vec{Q}$, S_D is positive definite on $\vec{Q}$ for $\delta > 0$. The assertion follows from the Sylvester law of inertia [24]. □

Remark 2.1 We employ a preconditioned conjugate gradient method to solve equation (2.7). The computation procedure of the CG method consists of scalar-vector multiplication, matrix-vector multiplication, vector addition, and inner product of vectors. Therefore it need not know exactly each entry of the matrix, in fact, it is sufficient to get the values of matrix-vector products. Of course, we can get each element of an $(n_V + n_Q) \times (n_V + n_Q)$ matrix from the matrix $\mathcal{A}$ with an appropriate basis of $R(\mathcal{A})$. We note that the obtained matrix is regular and we can use a direct method to solve such a linear system with the matrix. However, that procedure is very costly and not suitable for large-scale computation from the viewpoint of both memory requirements and computation speed. In our approach, the stiffness matrix is generated on all finite element nodes and operations of orthogonal projection are performed by procedures of inner product. Therefore, our solver is simple and easy to implement. Efficiency in the large-scale computing will be shown in Sections 2.5.2 and 3.7.

Remark 2.2 We note that the generalized version of a finite element solver in conventional codes, which is described as Algorithm A.1 in Appendix A.1, is applicable to the finite element equation of the *Dirichlet Problem*, but it is not applicable to the *Slip Problem*. Algorithm A.1 is feasible under an assumption on the dimension of the stiffness matrix. In the *Slip Problem*, this condition is not satisfied due to the following reason. To satisfy the assumption, surface integrations of outer normal components of finite element base functions should vanish. The spherical shell domain is approximated by a polyhedral domain in the *Slip Problem*. Hence, the integration is considered on the surface of the polyhedron that differs from the smooth surface of the spherical shell domain. Due to this perturbation caused by the approximation of the domain, the surface integrations of outer normal components of the finite element base functions do not vanish. Details are shown in Appendix A.2.

2.3 Preconditioned conjugate gradient solver for the Stokes equations

2.3.1 Preconditioned CG solver for indefinite linear systems

In this section we omit the symbol "$\vec{}$" from the vector and the subscript $\mathbb{R}^N$ from the inner product of vectors, i.e., we write u in place of $\vec{u}$ and $(\cdot, \cdot)$ in place of $(\cdot, \cdot)_{\mathbb{R}^N}$.

Let A be an $N \times N$ real symmetric matrix with $\dim R(A) = M$, where $0 < M \leq N$. We note that A may be indefinite. i.e., it may have positive, zero, and negative eigenvalues. Let $b \in R(A)$. We consider the problem to find $x \in R(A)$ satisfying

$$Ax = b\,. \tag{2.8}$$

There exists a unique solution $x = A^\dagger b \in R(A)$. Here $A^\dagger$ is the pseudo-inverse (the Moore–Penrose inverse) of A . Let P be the orthogonal projection of $\mathbb{R}^N$ onto $R(A)$,

$$(Pu,\, v) = (u,\, v) \quad \text{for all } v \in R(A)\,.$$

Let Q be an $N \times N$ real symmetric matrix with $R(Q) = R(A)$. We call Q '*preconditioner*'. It is desirable that QA approximates P and AQ approximates P in real computations. We study an algorithm to solve the linear system (2.8) with preconditioner Q.

Algorithm 2.1 *PCG Algorithm*

$$\begin{aligned}
&x_0 \in R(A)\,,\\
&r_0 := b - Ax_0\,,\\
&p_0 := Qr_0\,,\\
&\text{do } n = 0, 1, \ldots, \text{until } r_n = 0\\
&\quad \alpha_n := (Qr_n,\, r_n)/(A\,p_n,\, p_n)\,,\\
&\quad x_{n+1} := x_n + \alpha_n p_n\,,\\
&\quad r_{n+1} := r_n - \alpha_n A p_n\,,\\
&\quad \beta_n := (Qr_{n+1},\, r_{n+1})/(Qr_n,\, r_n)\,,\\
&\quad p_{n+1} := Qr_{n+1} + \beta_n p_n\,,\\
&\text{enddo.}
\end{aligned}$$

Usually the preconditioned CG method is used to solve a linear system with a positive definite matrix. A preconditioning matrix Q is also selected as a positive definite matrix. The convergence of the CG method for the positive definite matrix is shown in many text books, e.g. in [2, 24]. For the linear system with the symmetric matrix including positive and zero eigenvalues, behavior of the convergence of the CG method is reported in [32]. For the linear system with the symmetric matrix

including positive and negative eigenvalues, the SYMMLQ and MINRES methods are developed [34], where preconditioners with positive definite matrices are applicable. Algorithm 2.1 restricted to the symmetric positive definite matrices A and Q is found in [3]. However, in our study, we apply the CG method with an indefinite symmetric preconditioner to a linear system with an indefinite symmetric matrix. The idea of use of indefinite matrix as a preconditioner is found in the paper on the CSBCG method [4].

The preconditioned CG method for a general symmetric matrix is regarded as a variant of the bi-conjugate gradient (BiCG) method [20]. We recall the BiCG method to solve $Ax = b$.

Algorithm 2.2 *BiCG Algorithm*

x_0 : initial vector ,
$r_0 := b - Ax_0$,
$r_0^\star$: initial '*shadow*' residual ,
$p_0 := r_0$,
$p_0^\star := r_0^\star$,
do $n = 0, 1, \ldots,$ until $r_n = 0$
  $\alpha_n := (r_n,\, r_n^\star)/(A\,p_n,\, p_n^\star)$,
  $x_{n+1} := x_n + \alpha_n p_n$,
  $r_{n+1} := r_n - \alpha_n A p_n$,
  $r_{n+1}^\star := r_n^\star - \alpha_n A^T p_n^\star$,
  $\beta_n := (r_{n+1},\, r_{n+1}^\star)/(r_n,\, r_n^\star)$,
  $p_{n+1} := r_{n+1} + \beta_n p_n$,
  $p_{n+1}^\star := r_{n+1}^\star + \beta_n p_n^\star$,
enddo.

The procedure of the *PCG Algorithm* is equivalent to the BiCG method with selection of the initial '*shadow*' residual $r_0^\star = Q^\dagger r_0$, for the linear system

$$QAx = Qb\,.$$

It is clear from rewriting r_n by Qr_n and $p_n^\star$ by $Q^\dagger p_n^\star$.

The linear system is rewritten by using the initial vector x_0 and the initial residual r_0 as the problem to find $y \in R(A)$ satisfying

$$Ay = b - Ax_0 = r_0\,. \tag{2.9}$$

A solution $x_0 + y$ is equal to the solution of (2.8). The Krylov subspace is defined with A, b, and n by

$$K_n(A, b) := \mathrm{span}[b, Ab, \ldots, A^{n-1}b]\,.$$

Let N_1 be a number defined by

$$N_1(QA, Qr_0) := \min\{n\,;\, K_n(QA, Qr_0) = K_{n+1}(QA, Qr_0)\}\,.$$

Remark 2.3
(i) We note that

$$K_1(QA, Qr_0) \subset K_2(QA, Qr_0) \subset \ \cdots\ \subset K_{N_1}(QA, Qr_0) = K_{N_1+1}(QA, Qr_0) = \cdots$$

and $K_{N_1}(QA, Qr_0)$ is the largest Krylov subspace generated by QA and Qr_0.
(ii) We have the following relations in Krylov subspaces generated by AQ and r_0:

$$K_1(AQ, r_0) \subset K_2(AQ, r_0) \subset \ \cdots\ \subset K_{N_1}(AQ, r_0) = K_{N_1+1}(AQ, r_0) = \cdots\,.$$

Here $K_{N_1}(AQ, r_0)$ is the largest Krylov subspace.

Proposition 2.2 *There exist* $\{\gamma_i\}_{i=0}^{N_1-1}$ *with* $\gamma_0 \neq 0$, *satisfying*

$$(AQ)^{N_1} r_0 = \sum_{i=0}^{N_1-1} \gamma_i (AQ)^i r_0\,.$$

Proof. We will show it by a contradiction. We assume that $\gamma_0 = 0$. Then we would have

$$(AQ)^{N_1} r_0 = \sum_{i=1}^{N_1-1} \gamma_i (AQ)^i r_0\,.$$

Multiplying $Q^\dagger A^\dagger$ on both sides, we could get

$$(AQ)^{N_1-1} r_0 = \sum_{i=1}^{N_1-1} \gamma_i (AQ)^{i-1} r_0 \in K_{N_1-1}(AQ, r_0)\,.$$

This contradicts the fact that $K_{N_1}(AQ, r_0)$ is the largest Krylov subspace generated by AQ and r_0. □

Lemma 2.1 *A solution of* (2.9) *is found in the largest Krylov subspace:*

$$A^\dagger r_0 \in K_{N_1}(QA, Qr_0)\,.$$

Proof. Since $\gamma_0 \neq 0$, from Proposition 2.2, we have

$$r_0 = \frac{1}{\gamma_0}\left\{(AQ)^{N_1} r_0 - \sum_{i=1}^{N_1-1} \gamma_i (AQ)^i r_0\right\}\,.$$

Multiplying $A^\dagger$ to both sides, we obtain

$$\begin{aligned} A^\dagger r_0 &= \frac{1}{\gamma_0}\left\{(QA)^{N_1-1} Q r_0 - \sum_{i=1}^{N_1-1} \gamma_i (QA)^{i-1} Q r_0\right\} \\ &\in K_{N_1}(QA, Qr_0)\,. \ \square \end{aligned}$$

Lemma 2.2 *Suppose that matrix A be regular in $K_{N_1}(QA, Qr_0)$. Let $u \in K_{N_1}(QA, Qr_0)$ be a solution of the problem satisfying*

$$(Au - r_0,\, y) = 0 \quad \textit{for all } y \in K_{N_1}(QA, Qr_0)\,. \tag{2.10}$$

Then u satisfies

$$(Au - r_0,\, y) = 0 \quad \textit{for all } y \in R(A).$$

Proof. A vector $u_1 = A^\dagger r_0 \in K_{N_1}(QA, Qr_0)$ satisfies

$$(Au_1 - r_0,\, y) = 0 \quad \text{for all } y \in R(A).$$

From $K_{N_1}(QA, Qr_0) \subset R(A)$, we have

$$(Au_1 - r_0,\, z) = 0 \quad \text{for all } z \in K_{N_1}(QA, Qr_0).$$

Since the problem (2.10) has a unique solution, we have $u_1 = u$. □

The *PCG Algorithm* may stop without finding the solution of (2.9), $A^\dagger r_0$, due to division by zero. Since A and Q are indefinite, $(A\,p_n,\, p_n)$ and $(Q\,r_{n+1},\, r_{n+1})$ may vanish. We call such a situation '*breakdown*' of the *PCG Algorithm*. We are going to show a necessary and sufficient condition for no-breakdown of the *PCG Algorithm*.

Lemma 2.3 *Let n be a number equal to or smaller than $N_1(QA, Qr_0)$. Assume that for all number l $(0 \le l < n)$,*

$$(Ap_l,\, p_l) \neq 0 \textit{ and } (Qr_l,\, r_l) \neq 0\,.$$

Then for all number m $(1 \le m \le n)$ the followings hold:

(a) $(r_m,\, z) = 0 \quad \textit{for all } z \in K_m(QA, Qr_0)\,,$
(b) $(Ap_m,\, z) = 0 \quad \textit{for all } z \in K_m(QA, Qr_0)\,,$
(c) $\mathrm{span}[Qr_0, Qr_1, \ldots Qr_m] = \mathrm{span}[p_0, p_1, \ldots p_m] = K_{m+1}(QA, Qr_0)\,.$

Proof. We will prove it by induction. When $n = 1$, we can show (a), (b), and (c) as follows. We first obtain (a),

$$\begin{aligned}
(r_1,\, Qr_0) &= (r_0 - \alpha_0 Ap_0,\, Qr_0) \\
&= (r_0,\, Qr_0) - \alpha_0 (Ap_0,\, Qr_0) \\
&= (Qr_0,\, r_0) - \frac{(Qr_0,\, r_0)}{(Ap_0,\, p_0)} (Ap_0,\, p_0) \\
&= 0\,.
\end{aligned}$$

Second, we obtain (b), by virtue of symmetry of both A and Q,

$$\begin{aligned}(Ap_1,\, Qr_0) &= (Qr_1 + \beta_0 p_0,\, Ap_0) \\ &= (Qr_1 + \beta_0 p_0,\, \frac{1}{\alpha_0}(r_0 - r_1)) \\ &= \frac{1}{\alpha_0}(Qr_1,\, r_0) - \frac{\beta_0}{\alpha_0}(p_0,\, r_1) \\ &= \frac{1}{\alpha_0}(r_1,\, Qr_0) - \frac{\beta_0}{\alpha_0}(r_1,\, Qr_0) \\ &= 0\,.\end{aligned}$$

Here $\alpha_0 \neq 0$ from the assumption with $n = 1$, i.e., $(Ap_0,\, p_0) \neq 0$ and $(Qr_0,\, r_0) \neq 0$. Finally, we note that (c) follows from definitions of r_1, p_0, and p_1, and that $\alpha_0 \neq 0$.

We assume that when $n = k$, (a), (b), and (c) are valid. Since (c) holds when $m = k$, any $z \in K_{k+1}(QA, Qr_0)$ is expressed by

$$z = z_0 + \gamma_1 p_k\,,$$

where $z_0 \in K_k(QA, Qr_0)$ and $\gamma_1 \in \mathbb{R}$. We first show that (a) holds for $m = k + 1$. We note that r_{k+1} is orthogonal to both z_0 and p_k. In fact, we have, by virtue of (a) and (b) when $m = k$,

$$\begin{aligned}(r_{k+1},\, z_0) &= (r_k - \alpha_k A p_k,\, z_0) \\ &= (r_k,\, z_0) - \alpha_k (Ap_k,\, z_0) \\ &= 0\,.\end{aligned}$$

Further, we have, by virtue of (a) when $m = k$ and (c) when $m = k - 1$,

$$\begin{aligned}(r_{k+1},\, p_k) &= (r_k,\, p_k) - \alpha_k (Ap_k,\, p_k) \\ &= (r_k,\, p_k) - \frac{(Qr_k,\, r_k)}{(Ap_k,\, p_k)}(Ap_k,\, p_k) \\ &= (r_k,\, Qr_k + \beta_{k-1} p_{k-1}) - (Qr_k,\, r_k) \\ &= \beta_{k-1}(r_k,\, p_{k-1}) \\ &= 0\,.\end{aligned}$$

Second, we show that (b) holds for $m = k + 1$. We note that Ap_{k+1} is orthogonal to both z_0 and p_k. In fact, we have, by virtue of (a) when $m = k + 1$ and (b) when $m = k$,

$$\begin{aligned}(Ap_{k+1},\, z_0) &= (A(Qr_{k+1} + \beta_k p_k),\, z_0) \\ &= (r_{k+1},\, QAz_0) + \beta_k (Ap_k,\, z_0) \\ &= 0\,.\end{aligned}$$

Further, we have, by virtue of (a) when $m = k+1$ and (c) when $m = k$,

$$\begin{aligned}(Ap_{k+1},\, p_k) &= (Qr_{k+1},\, Ap_k) + \beta_k(Ap_k,\, p_k) \\ &= (Qr_{k+1},\, \frac{1}{\alpha_k}(r_k - r_{k+1})) + \beta_k(Ap_k,\, p_k) \\ &= \frac{1}{\alpha_k}(r_{k+1},\, Qr_k) - \frac{(Ap_k,\, p_k)}{(Qr_k,\, r_k)}(Qr_{k+1},\, r_{k+1}) + \frac{(Qr_{k+1},\, r_{k+1})}{(Qr_k,\, r_k)}(Ap_k,\, p_k) \\ &= 0\,.\end{aligned}$$

Finally, we note that (c) follows from definitions of r_{m+1} and p_{m+1} and that $\alpha_m \neq 0$. □

Lemma 2.4 *Let n be a number that satisfies $0 \leq n < N_1$. Then following two statements are equivalent.*

(a) $(Ap_l,\, p_l) \neq 0$ *and* $(Qr_l,\, r_l) \neq 0 \quad (0 \leq \forall l \leq n)\,.$

(b) *A is regular in $K_{l+1}(QA, Qr_0)$ and Q is regular in $K_{l+1}(AQ, r_0) \quad (0 \leq \forall l \leq n)\,.$*

Proof. We assume that (a) holds. Then, we have A is regular in $K_{l+1}(QA, Qr_0)$. Since $K_{l+1}(QA, Qr_0) = \mathrm{span}[p_0, p_1, \ldots, p_l]$, any $y \in K_{l+1}(QA, Qr_0)$ is expressed by

$$y = \sum_{j=0}^{l} \alpha_j p_j\,.$$

From Lemma 2.3, we have

$$(Ay,\, p_k) = (A\sum_{j=0}^{l} \alpha_j p_j,\, p_k) = \sum_{j=0}^{l} \alpha_j (Ap_j,\, p_k) = \alpha_k (Ap_k,\, p_k)\,,$$

for all k $(0 \leq k \leq l)$. By the assumption $(Ap_k,\, p_k) \neq 0$ for all k $(0 \leq k \leq l)$, we get $y = 0$ as a solution of the problem to find vector $y \in K_{l+1}(QA, Qr_0)$ satisfying

$$(Ay,\, z) = 0 \quad \text{for all } z \in K_{l+1}(QA, Qr_0)\,.$$

Therefore, we can obtain that A is regular in $K_{l+1}(QA, Qr_0)$. The regularity of Q in $K_{l+1}(AQ, r_0)$ is obtained in the same manner.

We assume that (b) holds. We show that (a) holds by induction. When $n = 0$, by definition of $p_0 = Qr_0$, we have $(Ap_0, p_0) \neq 0$. For $0 \leq l \leq n$, we assume that (a) and (b) are equivalent, and for $l = n+1$, (b) holds. $(Ap_{n+1},\, p_{n+1}) = 0$ leads to a contradiction as follows. Any $z \in K_{n+2}(QA, Qr_0)$ is expressed by

$$z = z_0 + \gamma_1 p_{n+1}\,,$$

where $z_0 \in K_{n+1}(QA, Qr_0)$ and $\gamma_1 \in \mathbb{R}$. By Lemma 2.3 we would get

$$(Ap_{n+1},\, z) = 0 \quad \text{for all } z \in K_{n+2}(QA, Qr_0)\,.$$

Since A is regular in $K_{n+2}(QA, Qr_0)$, we would obtain that $p_{n+1} = 0$. This contradicts (c) with $m = n+1$ in Lemma 2.3. We also find $(Qr_l,\, r_l) \neq 0$ in the same manner. □

Following Saad [39] and Joubert [31], we define moment matrices to describe a necessary and sufficient condition for no-breakdown of the *PCG Algorithm*.

Definition 2.1 *Let m be a number $(1 \leq m \leq N_1)$. We define $m \times m$ moment matrices $\mathcal{A}_m$ and $\mathcal{B}_m$ by*

$$[\mathcal{A}_m]_{i\,j} := ((QA)^{i+j-1}Qr_0,\, r_0) \quad (1 \leq i, j \leq m)\,,$$
$$[\mathcal{B}_m]_{i\,j} := ((QA)^{i+j-2}Qr_0,\, r_0) \quad (1 \leq i, j \leq m)\,.$$

Let $a_m := \det \mathcal{A}_m$ and $b_m := \det \mathcal{B}_m$.

Proposition 2.3 *Let m be a number $(1 \leq m \leq N_1)$.*

(i) *$a_m \neq 0$ if and only if A is regular in $K_m(QA, Qr_0)$,*

(ii) *$b_m \neq 0$ if and only if Q is regular in $K_m(AQ, r_0)$.*

Proof. Since $\{(QA)^{j-1}Qr_0\}_{j=1}^m$ is a basis of $K_m(QA, Qr_0)$ for $1 \leq m \leq N_1$, we have

$$\det[Qr_0, (QA)Qr_0, \ldots, (QA)^{m-1}Qr_0]^T A[Qr_0, (QA)Qr_0, \ldots, (QA)^{m-1}Qr_0] = \det\left[\{((QA)^{i+j-1}Qr_0, r_0)\}_{1\leq i,j\leq m}\right]\,.$$

This relation means (i). We also obtain (ii) in the same manner. □

We get a necessary and sufficient condition for no-breakdown of the *PCG Algorithm*.

Theorem 2.1 *Let $x_0 \in R(A)$. Suppose that $a_m \neq 0$ and $b_m \neq 0$ for all m $(1 \leq m \leq N_1)$. Then the PCG Algorithm does not break down and x_{N_1} is a solution of $Ax = b$ in $R(A)$,*

$$x_{N_1} = A^\dagger(b - A\,x_0) + x_0\,.$$

A sufficient condition of no-breakdown of the BiCG method is known.

Theorem 2.2 (Joubert [31]) *Let A be an $N \times N$ real regular matrix with at least one real eigenvalue. Let N_0 be a number defined by $N_0(A, y) = \min\{n\,;\, K_n(A, y) = K_{n+1}(A, y)\}$. Let k be a positive integer. Then*

$$\det[y, A^T y, \ldots (A^T)^{m-1}y]^T A^k[y, Ay, \ldots A^{m-1}y] \neq 0$$

for almost every vector $y \in \mathbb{R}^N$, where $1 \leq m \leq N_0$.

Theorem 2.3 *Let Q be positive definite on $R(Q)$. Then for almost every vector $x_0 \in R(A)$, the PCG Algorithm does not break down.*

Proof. There exists a Cholesky decomposition $Q = W\,W^T$, where W is a real matrix and $R(W) = R(Q)$. We have

$$\begin{aligned}[\mathcal{A}_m]_{i\,j} &= ((W\,W^T A)^{i+j-1} W\,W^T r_0,\, r_0) \\ &= ((W^T AW)^{i+j-1} W^T r_0,\, W^T r_0)\,.\end{aligned}$$

Since $W^T AW$ is real symmetric, all eigenvalues of $W^T AW$ are real. Using Theorem 2.2, we obtain the desired result. □

2.3.2 Convergence history of preconditioned CG solver of the Stokes equations

In this section, we will observe the convergence of the *PCG Algorithm* for the finite element equations of the *Dirichlet Problem* by a numerical experiment.

In the *PCG Algorithm*, we have three requirements for the preconditioner Q. The first requirement is that $R(Q) = R(A)$, and the second is that both QA and AQ should approximate P. The third is the following. Construction of Q and multiplication of Q by a vector should be performed with low computation cost. Incomplete Cholesky decomposition [24] is widely used as a perconditioner for the CG method to solve a linear system with a positive definite matrix, because it meets the second and third requirements. We add the orthogonal projection to a matrix of incomplete Cholesky decomposition operation to meet the first requirement.

We recall the finite element equation of the Stokes problem with the Dirichlet boundary condition (2.7), to find $(\vec{u}_0, \vec{p}) \in \vec{V} \times \vec{Q}$ satisfying

$$\mathcal{P}\mathcal{A}_0\mathcal{P}\begin{pmatrix}\vec{u}_0\\ \vec{p}\end{pmatrix} = \mathcal{P}\begin{pmatrix}\vec{f}_1 - A\vec{u}_g\\ \vec{f}_2 - B\vec{u}_g\end{pmatrix},$$

where $\mathcal{A}_0 := \begin{pmatrix}A & B^T\\ B & -\delta D\end{pmatrix}$, $\mathcal{P} := \begin{pmatrix}P_{\vec{V}} & 0\\ 0 & P_{\vec{Q}}\end{pmatrix}$, and $\vec{u}_g$ belongs to $\vec{V}(g)$.

We present a practical version of the *PCG Algorithm* for the problem (2.8).

Algorithm 2.3

$x_0 \in R(A)$,
$r_0 := P(b - Ax_0)$,
$p_0 := Qr_0$,
do $n = 0, 1, \ldots,$ until $r_n = 0$
  $\alpha_n := (Qr_n,\, r_n)/(A\,p_n,\, p_n)$,
  $x_{n+1} := P(x_n + \alpha_n p_n)$,
  $r_{n+1} := P(r_n - \alpha_n A p_n)$,
  $\beta_n := (Qr_{n+1},\, r_{n+1})/(Qr_n,\, r_n)$,
  $p_{n+1} := P(Qr_{n+1} + \beta_n p_n)$,
enddo.

Here we apply projection P to updated vectors to keep them in $R(A)$. In case of the Stokes problem, $R(A)$ is equal to the solution space $\vec{V} \times \vec{Q}$. By this procedure we can avoid instability caused by accumulations of round-off errors.

We consider two types of preconditioners:

$$Q_1 := \mathcal{P} \begin{pmatrix} \widetilde{A}^{-1} & 0 \\ 0 & \frac{1}{\delta}\widetilde{D}^{-1} \end{pmatrix} \mathcal{P}\,,$$
$$Q_2 := \mathcal{P}\widetilde{\mathcal{A}_0}^{-1}\mathcal{P}\,,$$

where $\widetilde{M}$ stands for an incomplete modified Cholesky decomposition of a matrix M, i.e., a product of a lower triangular matrix L_M and a diagonal matrix D_M is close to the original matrix M,

$$L_M D_M {L_M}^T = \widetilde{M} \sim M\,.$$

We note that Q_1 is positive definite on $\vec{V} \times \vec{Q}$ and Q_2 is indefinite. We have more freedom in construction of preconditioner in the class of indefinite symmetric matrices than in the class of positive definite matrices.

We consider a test problem of the *Dirichlet Problem*, and set $\Omega := \{x \in \mathbb{R}^3\,;\, 0.5 < |x| < 1\}$. Suppose it has the solution:

$$u = \begin{pmatrix} \sin x_1 - x_1 \cos x_2, \\ 2(\sin x_2 - x_2 \cos x_3), \\ 2\sin x_3 - x_3(\cos x_2 + \cos x_1) \end{pmatrix}\,,$$
$$p = \sin x_1 + \sin x_2 + \sin x_3 + c\,.$$

Dirichlet data are naturally chosen to be compatible with the solution. We employ the stabilized finite element method of the penalty type (1.41) and set the stability parameter δ to be 0.1.

Figure 2.1 shows a mesh of tetrahedra of a spherical shell domain with 117,540 nodes and 664,320 elements. An eighth part of the shell is cut off to show an interior mesh. Table 2.1 shows degrees of freedom of the problem obtained from discretization using the mesh. Table 2.2 shows an error of the finite element solution. Figures 2.2 and 2.3 show convergence histories of the *PCG Algorithm* with preconditioner Q_1 and Q_2, respectively. In both cases of preconditioners, the *PCG Algorithm* converges without breakdown. Some oscillation of residuals is observed in both convergence histories. We note that, in the CG method for indefinite matrices without any preconditioner, the convergence with similar oscillation is reported [35]. After 200 times iteration we have obtained a numerical solution with relative residual of 10^{-8}. We can conclude that Algorithm 2.3 is very efficient to solve the Stokes equations.

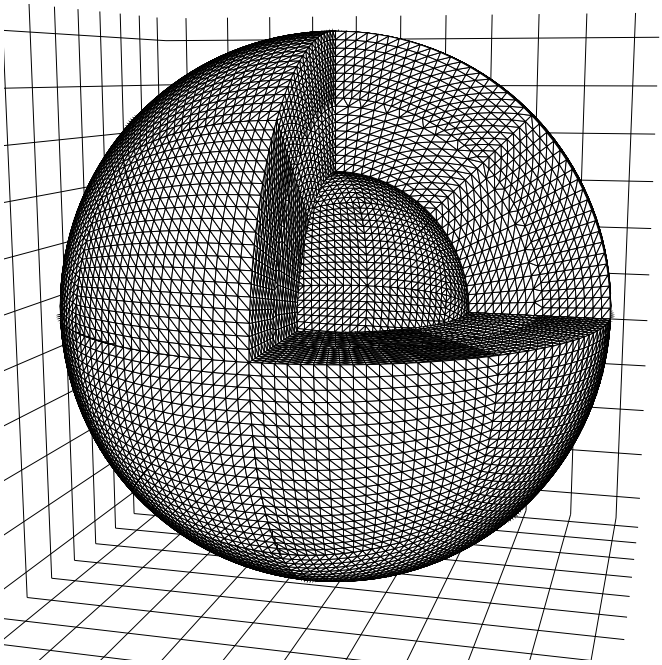

Figure 2.1. Mesh of tetrahedra of a spherical shell domain with 117,540 nodes and 664,320 elements

Table 2.1. Degrees of freedom of the Stokes problem

n_X	n_S	$n_X + n_S$	$n_V + n_Q$
352,620	117,540	470,160	436,979

Table 2.2. Error of FEM solution

h	$\|\|u - u_h\|\|_1 / \|\|u\|\|_1$	$\|\|p - p_h\|\|_0 / \|\|p\|\|_0$
7.5207×10^{-2}	4.0151×10^{-2}	1.8363×10^{-2}

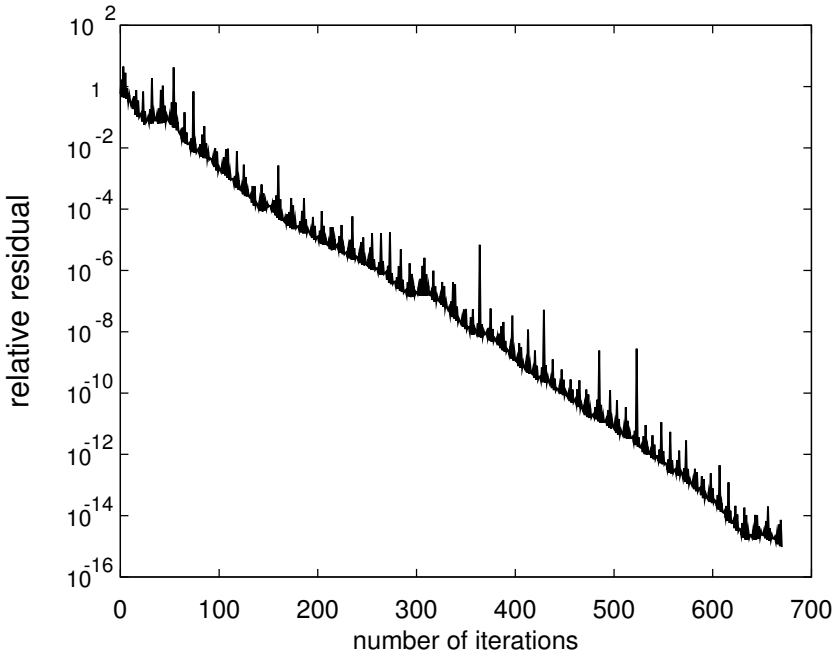

Figure 2.2. Convergence history of the *PCG Algorithm* with preconditioner Q_1

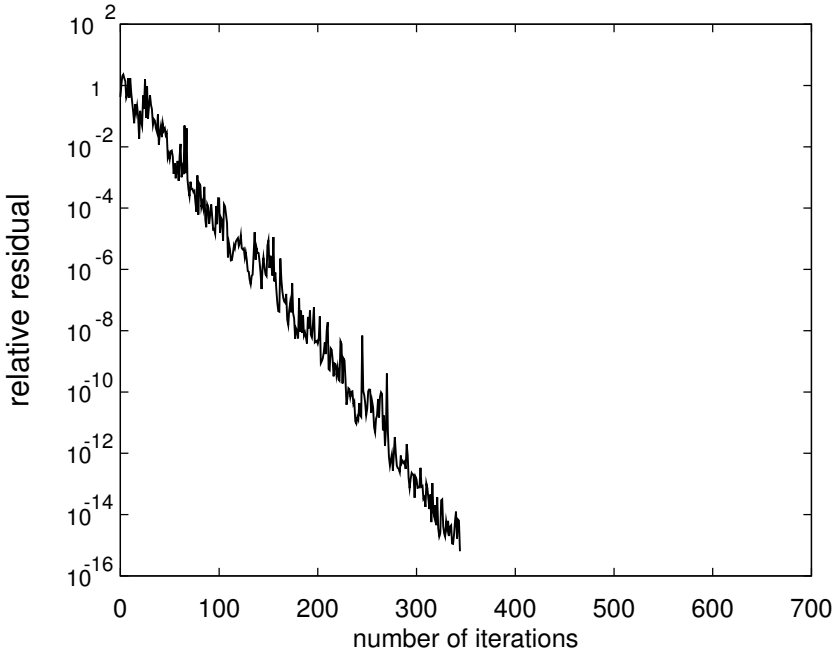

Figure 2.3. Convergence history of the *PCG Algorithm* with preconditioner Q_2

2.4 Domain decomposition into congruent subdomains

In this section, we present a method to reduce memory requirements for finite element matrices, which consists of domain decomposition into congruent subdomains.

2.4.1 A class of orthogonal transformations and domain decomposition into congruent subdomains

Let R be an orthogonal matrix that is a product of a diagonal matrix $S = \mathrm{diag}\{\epsilon_k\}_{1\le k\le 3}$ with $\epsilon_k = 1$ or -1 and a permutation matrix P,

$$R = S\,P\,. \tag{2.11}$$

We call S '*sign matrix*'. Let F be an affine transformation in $\mathbb{R}^3$ defined by

$$F(x) := R\,x + c\,,$$

where $c \in \mathbb{R}^3$ is a displacement vector. Let Ω be a domain in $\mathbb{R}^3$ and Ω' be the image of Ω by F. Let p and u be scalar- and vector-valued functions of $x \in \Omega$, respectively. We define the transformed scalar- and vector-valued functions of $x' := F(x) \in \Omega'$ by

$$\begin{aligned} p' &:= p \circ F^{-1}\,, \\ u' &:= R(u \circ F^{-1})\,. \end{aligned}$$

Let ∇' be the nabla operator in x' and $D'(u')$ be the tensor corresponding to $D(u)$.

Lemma 2.5 *We have*

$$\nabla' p' = R\,\nabla p\,, \tag{2.12}$$
$$\nabla' \cdot u' = \nabla \cdot u\,, \tag{2.13}$$
$$D'(u') = R\,D(u)R^{-1}\,, \tag{2.14}$$
$$D'(u') : D'(v') = D(u) : D(v)\,. \tag{2.15}$$

Recall that

$$D(u) : D(v) = \sum_{1\le i,j\le 3} [D(u)]_{i\,j}[D(v)]_{i\,j}\,.$$

We decompose the domain Ω into p non-overlapping subdomains $\{\Omega^{(i)}\}_{0\le i<p}$,

$$\bar{\Omega} = \bigcup_{0\le i<p} \bar{\Omega}^{(i)}, \quad \Omega^{(i)} \cap \Omega^{(j)} = \emptyset \quad (0 \le i < j < p)\,.$$

Here $\bar{\Omega}$ means the closure of Ω. We call $\Omega^{(0)}$ '*reference subdomain*'.

Assumption 2.1 *Every $\Omega^{(i)}$ $(i = 1, 2, \ldots, p-1)$ is an image of the reference subdomain $\Omega^{(0)}$ by an affine transformation $F^{(i)}$ with an orthogonal matrix $R^{(i)} = S^{(i)}P^{(i)}$ and a displacement $c^{(i)} \in \mathbb{R}^3$. Here $S^{(i)}$ is a sign matrix,*

$$[S^{(i)}]_{k\,l} := \delta_{k\,l}\epsilon_k^{(i)}\,,$$

and $P^{(i)}$ is a permutation matrix.

We set the sign matrix $S^{(0)}$ and the permutation matrix $P^{(0)}$ to be equal to the identity matrix.

Let $\mathcal{T}_h^{(0)}$ be the partition of $\bar{\Omega}^{(0)}$ by tetrahedra. Let $\mathcal{T}_h^{(i)}$ be the partition of $\bar{\Omega}^{(i)}$, which is obtained by the partition $\mathcal{T}_h^{(0)}$ and the affine transformation $F^{(i)}$ as follows

$$\mathcal{T}_h^{(i)} = \{K' \in \mathcal{T}_h\ ;\ K' = F^{(i)}K,\ \forall\, K \in \mathcal{T}_h^{(0)}\} \quad (1 \le i < p)\,.$$

Assumption 2.2 *A union of partitions $\bigcup_{i=0}^{p-1} \mathcal{T}_h^{(i)}$ is a partition of $\bar{\Omega}$.*

We suppose that a domain Ω is decomposable into subdomains satisfying Assumption 2.1 and a partition of the domain satisfies Assumption 2.2.

Let $\Lambda_G^{(i)} \subset \Lambda_G$ $(\#\Lambda_G^{(i)} = n_G^{(0)})$ be an index set of the nodes in $\bar{\Omega}^{(i)}$ for $i = 0, 1, \ldots, p-1$. We assume that $\Lambda_G^{(0)} := \{1, 2, \ldots, n_G^{(0)}\}$. We define a bijection $f^{(i)}$ from $\Lambda_G^{(0)}$ onto $\Lambda_G^{(i)}$,

$$f^{(i)}(\mu^{(0)}) = \mu^{(i)} \qquad (\mu^{(0)} \in \Lambda_G^{(0)},\ \mu^{(i)} \in \Lambda_G^{(i)})$$

by $x(P_{\mu^{(i)}}) = F^{(i)}x(P_{\mu^{(0)}})$. The bijection $f^{(0)}$ is identical on $\Lambda_S^{(0)}$. We define an index set corresponding to the ith subdomain,

$$\begin{aligned}
&\Lambda_S^{(i)} := \Lambda_G^{(i)} \quad (0 \le i < p)\,,\\
&\Lambda_X^{(i)} := \{\alpha \in \Lambda_X\ ;\ \alpha = [\alpha_0,\, k],\ \alpha_0 \in \Lambda_G^{(i)},\ k \in \{1, 2, 3\}\} \quad (0 \le i < p).
\end{aligned}$$

We note that $\#\Lambda_S^{(i)} = n_G^{(0)}$ and $\#\Lambda_X^{(i)} = 3\,n_G^{(0)}$ for $i = 0, 1, \ldots, p-1$, and $\Lambda_X^{(0)} = \{1, 2, \ldots, n_X^{(0)} (= 3\,n_G^{(0)})\}$. We define a bijection $f_X^{(i)}$ from $\Lambda_X^{(0)}$ onto $\Lambda_X^{(i)}$ by

$$f_X^{(i)}(\alpha) = [f^{(i)}(\alpha_0^{(0)}),\, \sigma^{(i)}(k^{(0)})] = [\alpha_0^{(i)},\, k^{(i)}] \quad (\alpha^{(0)} = [\alpha_0^{(0)},\, k^{(0)}] \in \Lambda_X^{(0)})\,,$$

for $i = 0, 1, \ldots, p-1$. Here $\sigma^{(i)}$ is a permutation defining $P^{(i)}$,

$$[P^{(i)}]_{\sigma^{(i)}(k)\,k} = 1\,.$$

We note that

$$[R^{(i)}]_{\sigma^{(i)}(k)\,l} = \epsilon^{(i)}_{\sigma^{(i)}(k)}\delta_{k\,l} \quad (k, l \in \{1, 2, 3\})\,.$$

2.4.2 Finite element bases and matrices in subdomains

We have relations between a finite element base function in the reference subdomain and that in transformed subdomains in both scalar- and vector-valued cases. We recall that $\{\phi_\mu\}_{\mu\in\Lambda_S}$ is a scalar-valued finite element basis of S_h and $\{\varphi_\alpha\}_{\alpha\in\Lambda_X}$ is a vector-valued finite element basis of X_h.

Lemma 2.6 *Let $\mu^{(i)} = f^{(i)}(\mu^{(0)})$ for $\mu^{(0)} \in \Lambda_S^{(0)}$ and $\alpha^{(i)} = [\alpha_0^{(i)}, k^{(i)}] = f_X^{(i)}(\alpha^{(0)})$ for $\alpha^{(0)} = [\alpha_0^{(0)}, k^{(0)}] \in \Lambda_X^{(0)}$, respectively. Then we have*

$$
\begin{aligned}
\psi_{\mu^{(i)}} &= \psi_{\mu^{(0)}} \circ {F^{(i)}}^{-1}\,, \\
\varphi_{\alpha^{(i)}} &= \epsilon^{(i)}_{k^{(i)}} R^{(i)}(\varphi_{\alpha^{(0)}} \circ {F^{(i)}}^{-1})\,.
\end{aligned}
$$

Proof. The relation between the scalar-valued base functions is trivial. We prove the relation between vector-valued base functions. We have

$$
\begin{aligned}
\Big[\epsilon^{(i)}_{k^{(i)}} R^{(i)}(\varphi_{\alpha^{(0)}} \circ {F^{(i)}}^{-1})\Big]_n &= \epsilon^{(i)}_{k^{(i)}} \sum_{1\le m\le 3} [R^{(i)}]_{n\,m} [\varphi_{[\alpha_0^{(0)},\,k^{(0)}]} \circ {F^{(i)}}^{-1}]_m \\
&= \sum_{1\le m\le 3} [R^{(i)}]_{n\,m} \epsilon^{(i)}_{k^{(i)}} \delta_{k^{(0)}\,m} (\psi_{\alpha_0^{(0)}} \circ {F^{(i)}}^{-1}) \\
&= \sum_{1\le m\le 3} [R^{(i)}]_{n\,m} [R^{(i)}]_{k^{(i)}\,m} (\psi_{\alpha_0^{(0)}} \circ {F^{(i)}}^{-1}) \\
&= \delta_{n\,k^{(i)}} \psi_{\alpha_0^{(i)}} \\
&= \Big[\varphi_{[\alpha_0^{(i)},\,k^{(i)}]}\Big]_n\,,
\end{aligned}
$$

for $n = 1, 2, 3$. □

Let $A^{(i)}$, $B^{(i)}$, $D^{(i)}$, $M_1^{(i)}$, and $M_0^{(i)}$ be stiffness sub-matrices and mass sub-matrices in $\Omega^{(i)}$ defined by

$$
\begin{aligned}
[A^{(i)}]_{\alpha^{(i)}\,\beta^{(i)}} &:= a(\varphi_{\beta^{(i)}}, \varphi_{\alpha^{(i)}}\,;\,\Omega^{(i)}) && (\alpha^{(i)}, \beta^{(i)} \in \Lambda_X^{(i)})\,, && (2.16a)\\
[B^{(i)}]_{\mu^{(i)}\,\beta^{(i)}} &:= b(\varphi_{\beta^{(i)}}, \psi_{\mu^{(i)}}\,;\,\Omega^{(i)}) && (\mu^{(i)} \in \Lambda_S^{(i)},\ \beta^{(i)} \in \Lambda_X^{(i)})\,, && (2.16b)\\
[M_1^{(i)}]_{\alpha^{(i)}\,\beta^{(i)}} &:= (\varphi_{\beta^{(i)}}, \varphi_{\alpha^{(i)}}\,;\,\Omega^{(i)}) && (\alpha^{(i)}, \beta^{(i)} \in \Lambda_X^{(i)})\,, && (2.16c)\\
[D^{(i)}]_{\mu^{(i)}\,\nu^{(i)}} &:= d(\psi_{\nu^{(i)}}, \psi_{\mu^{(i)}}\,;\,\Omega^{(i)}) && (\mu^{(i)}, \nu^{(i)} \in \Lambda_S^{(i)})\,, && (2.16d)\\
[M_0^{(i)}]_{\mu^{(i)}\,\nu^{(i)}} &:= (\psi_{\nu^{(i)}}, \psi_{\mu^{(i)}}\,;\,\Omega^{(i)}) && (\mu^{(i)}, \nu^{(i)} \in \Lambda_S^{(i)})\,. && (2.16e)
\end{aligned}
$$

We have relations on stiffness and mass matrices between the reference subdomain and other subdomains.

Theorem 2.4 *Let* $\alpha^{(0)} = [\alpha_0^{(0)}, k^{(0)}] \in \Lambda_X^{(0)}$, $\beta^{(0)} = [\beta_0^{(0)}, l^{(0)}] \in \Lambda_X^{(0)}$, *and* $\mu^{(0)}, \nu^{(0)} \in \Lambda_S^{(0)}$. *We have*

$$[A^{(i)}]_{\alpha^{(i)}\,\beta^{(i)}} = \epsilon^{(i)}_{k^{(i)}}\ [A^{(0)}]_{\alpha^{(0)}\,\beta^{(0)}}\ \epsilon^{(i)}_{l^{(i)}}, \tag{2.17a}$$

$$[B^{(i)}]_{\mu^{(i)}\,\beta^{(i)}} = [B^{(0)}]_{\mu^{(0)}\,\beta^{(0)}}\ \epsilon^{(i)}_{l^{(i)}}\,, \tag{2.17b}$$

$$[M_1^{(i)}]_{\alpha^{(i)}\,\beta^{(i)}} = [M_1^{(0)}]_{\alpha^{(0)}\,\beta^{(0)}}\,, \tag{2.17c}$$

$$[D^{(i)}]_{\mu^{(i)}\,\nu^{(i)}} = [D^{(0)}]_{\mu^{(0)}\,\nu^{(0)}}\,, \tag{2.17d}$$

$$[M_0^{(i)}]_{\mu^{(i)}\,\nu^{(i)}} = [M_0^{(0)}]_{\mu^{(0)}\,\nu^{(0)}}\,, \tag{2.17e}$$

where $\alpha^{(i)} = [\alpha_0^{(i)}, k^{(i)}] = f_X^{(i)}(\alpha^{(0)})$, $\beta^{(i)} = [\beta_0^{(i)}, l^{(i)}] = f_X^{(i)}(\beta^{(0)})$, $\mu^{(i)} = f^{(i)}(\mu^{(0)})$ *and* $\nu^{(i)} = f^{(i)}(\nu^{(0)})$.

Proof. We will prove the relation between the vector-valued stiffness matrices. We obtain

$$\begin{aligned}
[A^{(i)}]_{\alpha^{(i)}\,\beta^{(i)}} &= 2\int_{\Omega^{(i)}} D(\varphi_{\beta^{(i)}}(x)) : D(\varphi_{\alpha^{(i)}}(x))dx \\
&= 2\int_{\Omega^{(i)}} D(\epsilon^{(i)}_{l^{(i)}} R(\varphi_{\beta^{(0)}}(F^{-1}(x))) : D(\epsilon^{(i)}_{k^{(i)}} R(\varphi_{\alpha^{(0)}}(F^{-1}(x)))dx \\
&= \epsilon^{(i)}_{l^{(i)}}\epsilon^{(i)}_{k^{(i)}} 2\int_{\Omega^{(0)}} D(\varphi_{\beta^{(0)}}(x)) : D(\varphi_{\alpha^{(0)}}(x))|\mathrm{det}R|dx \\
&= \epsilon^{(i)}_{k^{(i)}} [A^{(0)}]_{\alpha^{(0)}\,\beta^{(0)}} \epsilon^{(i)}_{l^{(i)}}\,.
\end{aligned}$$

The relation between the vector-valued mass matrices is also obtained:

$$\begin{aligned}
[M_1^{(i)}]_{\alpha^{(i)}\,\beta^{(i)}} &= \epsilon^{(i)}_{l^{(i)}}\epsilon^{(i)}_{k^{(i)}} \int_{\Omega^{(0)}} \varphi_{[\beta_0^{(0)},\, l^{(0)}]}(x) \cdot \varphi_{[\alpha_0^{(0)},\, k^{(0)}]}(x)dx \\
&= \epsilon^{(i)}_{l^{(i)}}\epsilon^{(i)}_{k^{(i)}} \int_{\Omega^{(0)}} \sum_{1\le s\le 3} \delta_{l^{(0)}\,s}\psi_{\beta_0^{(0)}}(x)\delta_{k^{(0)}\,s}\psi_{\alpha_0^{(0)}}(x)dx \\
&= [M_1^{(0)}]_{[\alpha_0^{(0)},\, k^{(0)}]\,[\beta_0^{(0)},\, l^{(0)}]}\,.
\end{aligned}$$

Relations between the scalar-valued stiffness and mass matrices are trivial. □

Transformation of finite element matrices are given by renumbering indices and changing sings. We will introduce a strategy to restore values of the whole matrix from values of the sub-matrix in the reference subdomain.

For $\mu \in \Lambda_S$ and $\alpha \in \Lambda_X$ we define $I(\mu)$ and $I(\alpha)$ by

$$\begin{aligned}
I(\mu) &:= \{j \in \{0, 1, \ldots, p-1\}\ ;\ \mu \in f^{(j)}(\Lambda_S^{(0)})\}, \\
I(\alpha) &:= \{j \in \{0, 1, \ldots, p-1\}\ ;\ \alpha \in f_X^{(j)}(\Lambda_X^{(0)})\}.
\end{aligned}$$

For $\mu \in \Lambda_S$ and $j \in I(\mu)$ we define $\tilde{\mu}^{(j)} \in \Lambda_S^{(0)}$ as a unique element satisfying $f^{(j)}(\tilde{\mu}^{(j)}) = \mu$. Similarly for $\alpha \in \Lambda_X$ and $j \in I(\alpha)$ we define $\tilde{\alpha}^{(j)} \in \Lambda_X^{(0)}$ by $f_X^{(j)}(\tilde{\alpha}^{(j)}) = \alpha$.

Theorem 2.5 *Let* $\mu,\ \nu \in \Lambda_S$ *and* $\alpha = [\alpha_0,\, k],\ \beta = [\beta_0,\, l] \in \Lambda_X$. *We have*

$$[A]_{\alpha\,\beta} = \sum_{j \in I(\alpha) \cap I(\beta)} \epsilon_k^{(j)} [A^{(0)}]_{\tilde{\alpha}^{(j)}\,\tilde{\beta}^{(j)}} \epsilon_l^{(j)}\,, \tag{2.18a}$$

$$[B]_{\mu\,\beta} = \sum_{j \in I(\mu) \cap I(\beta)} [B^{(0)}]_{\tilde{\mu}^{(j)}\,\tilde{\beta}^{(j)}} \epsilon_l^{(j)}\,, \tag{2.18b}$$

$$[M_1]_{\alpha\,\beta} = \sum_{j \in I(\alpha) \cap I(\beta)} [M_1^{(0)}]_{\tilde{\alpha}^{(j)}\,\tilde{\beta}^{(j)}}\,, \tag{2.18c}$$

$$[D]_{\mu\,\nu} = \sum_{j \in I(\mu) \cap I(\nu)} [D^{(0)}]_{\tilde{\mu}^{(j)}\,\tilde{\nu}^{(j)}}\,, \tag{2.18d}$$

$$[M_0]_{\mu\,\nu} = \sum_{j \in I(\mu) \cap I(\nu)} [M_0^{(0)}]_{\tilde{\mu}^{(j)}\,\tilde{\nu}^{(j)}}\,. \tag{2.18e}$$

Proof. We prove only relation (2.18a). By definition of the stiffness matrix (2.3a), we have

$$\begin{aligned}
[A]_{\alpha\,\beta} &= \sum_{0 \le j < p} a(\varphi_\beta,\, \varphi_\alpha\,;\, \Omega^{(j)}) \\
&= \sum_{j \in I(\alpha) \cap I(\beta)} a(\varphi_\beta,\, \varphi_\alpha\,;\, \Omega^{(j)}) \\
&= \sum_{j \in I(\alpha) \cap I(\beta)} \epsilon_k^{(j)} [A^{(0)}]_{\tilde{\alpha}^{(j)}\,\tilde{\beta}^{(j)}} \epsilon_l^{(j)}\,.
\end{aligned}$$

Other relations are obtained similarly. □

Let $u \in \mathbb{R}^{n_X}$ be a given vector. The product of A and u is calculated in three steps. We prepare work-vectors $\{\vec{v}^{(i)}\}_{0 \le i < p},\ \{\vec{w}^{(i)}\}_{0 \le i < p} \subset \mathbb{R}^{n_X^{(0)}}$.

Algorithm 2.4

Step 1. *Decompose the vectors into sub-vectors with signs: for* $j = 0, 1, \ldots, p-1$,

$$[\vec{v}^{(j)}]_\delta := \epsilon_n^{(j)} [\vec{u}]_{f_X^{(j)}(\delta)} \quad (\delta = [\delta_0,\, n] \in \Lambda_X^{(0)})\,, \tag{2.19}$$

where $f_X^{(j)}(\delta) = \delta' \in \Lambda_X$.

Step 2. *Calculate matrix-vector products in subdomains : for* $j = 0, 1, \ldots, p-1$,

$$\vec{w}^{(j)} := A^{(0)}\, \vec{v}^{(j)}\,. \tag{2.20}$$

Step 3. *Change signs and sum up the values on the interfaces of subdomains :*

$$[A\,\vec{u}]_\alpha := \sum_{j \in I(\alpha)} \epsilon_k^{(j)} [\vec{w}^{(j)}]_{\tilde{\alpha}^{(j)}}\,, \tag{2.21}$$

where $\alpha = [\alpha_0,\, k] \in \Lambda_X$ *and* $\tilde{\alpha}^{(j)} \in \Lambda_X^{(0)}$.

In this algorithm we only need to store $A^{(0)}$ and $\{\epsilon_k^{(j)}\}_{k=1,2,3,j=0,\ldots,p-1}$. Therefore, memory requirement for a finite element solver is reduced drastically. The matrix-vector multiplication is a main operation in the execution of the CG method.

2.4.3 Efficient memory reduction in a spherical shell domain

We consider a spherical shell domain

$$\Omega := \{x \in \mathbb{R}^3 \ ; \ R_1 < |x| < R_2\} \, ,$$

which appears in the problem of Earth's mantle convection. Details of the numerical simulation of the problem are described in Section 3.

Proposition 2.4 *An available number of subdomains produced by transformations* (2.11) *is one of* $2, 3, 4, 6, 8, 12, 16, 24,$ *and* 48.
Proof. The number of combination of three nonzero entries of such matrices is 6 and the number of combination of signs at three entries is 2^3, because each entry can take only -1 or 1. Hence, the maximum number of the subdomains is 48. Examples of reference subdomains and generators of the orthogonal matrices are listed in Table 2.3. □

Figure 2.4 shows a partition of a spherical shell domain with data listed in Table 2.4 and a domain decomposition when $p = 48$. The reference subdomain is cut off and every subdomain is shifted slightly to show the decomposition clearly.

From the relation between total matrices and sub-matrices, the size of memory requirements in p subdomains would reduce to $1/p$ of that in a whole domain without decomposition. In reality, however, we must keep additional storage in renumbering the indices. We are going to see the practical reduction rates.

Since finite element matrices are sparse, it is efficient to store only nonzero components. We employ the Compressed Row Storage (CRS) format[3]. Let A be an $n_X \times n_X$ finite element matrix and n_A be a number of nonzero components of A. The CRS format uses three vectors $\vec{v}_A \in \mathbb{R}^{n_A}$, $\vec{c}_A \in \mathbb{N}^{n_A}$, and $\vec{r}_A \in \mathbb{N}^{n_X+1}$. Nonzero components of A are stored in $\vec{v}_A$, and information on rows and columns in $\vec{r}_A$ and $\vec{c}_A$, respectively. In detail, $[\vec{r}_A]_i$ is set to be the total number of nonzero components in rows of A with indices from 1 to $i-1$. Let $[A]_{ij}$ be kth nonzero component in ith row. Then we set $[\vec{v}_A]_l = [A]_{ij}$ and $[\vec{c}_A]_l = j$, where $l = [\vec{r}_A]_{i+k}$. In usual computation, 8 and 4 bytes are necessary to store a real number with double precision for values of the matrix and an integer for an index, respectively. Therefore, to store the matrix A by the CRS format, $12n_A + 4(n_X + 1)$ bytes are required.

Table 2.3. Examples of decompositions of a spherical shell domain into congruent subdomains (p : number of subdomains).

p	reference subdomain	generator of $R^{(i)}$
2	$\Omega_2^{(0)} := \{x \in \Omega \ ; \ x_3 > 0\}$	A_3
3	$\Omega_3^{(0)} := \{x \in \Omega \ ; \ x_1 > x_2, \ x_1 > x_3\}$	B
4	$\Omega_4^{(0)} := \{x \in \Omega \ ; \ x_2 > 0, \ x_3 > 0\}$	A_2, A_3
6	$\Omega_6^{(0)} := \{x \in \Omega \ ; \ x_1 > \lvert x_2\rvert, \ x_1 > \lvert x_3\rvert\}$	A_0, B
8	$\Omega_8^{(0)} := \{x \in \Omega \ ; \ x_i > 0 \ (i = 1, 2, 3)\}$	A_1, A_2, A_3
12	$\Omega_{12}^{(0)} := \{x \in \Omega \ ; \ x_1 > \lvert x_2\rvert, \ x_1 > \lvert x_3\rvert, \ x_2 > x_3\}$	A_0, B, C
16	$\Omega_{16}^{(0)} := \{x \in \Omega \ ; \ x_i > 0 \ (i = 1, 2, 3), \ x_2 > x_3\}$	A_1, A_2, A_3, C
24	$\Omega_{24}^{(0)} := \{x \in \Omega \ ; \ x_i > 0 \ (i = 1, 2, 3), \ x_1 > x_2, \ x_1 > x_3\}$	A_1, A_2, A_3, B
48	$\Omega_{48}^{(0)} := \{x \in \Omega \ ; \ x_i > 0 \ (i = 1, 2, 3), \ x_1 > x_2, \ x_1 > x_3, \ x_2 > x_3\}$	A_1, A_2, A_3, B, C

Matrices A_0, A_1, A_2, A_3, B, and C are defined by

$$A_0 = \begin{pmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{pmatrix}, \quad A_1 = \begin{pmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \quad A_2 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{pmatrix},$$

$$A_3 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{pmatrix}, \quad B = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}, \quad C = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}.$$

Table 2.4. Mesh data of a spherical shell domain (n_G : numbers of nodes, n_E : numbers of elements, n_S : degrees of freedom of scalar-valued unknowns, n_X : degrees of freedom of scalar-valued unknowns)

n_G	n_E	n_S	n_X
324,532	1,868,544	324,532	973,596

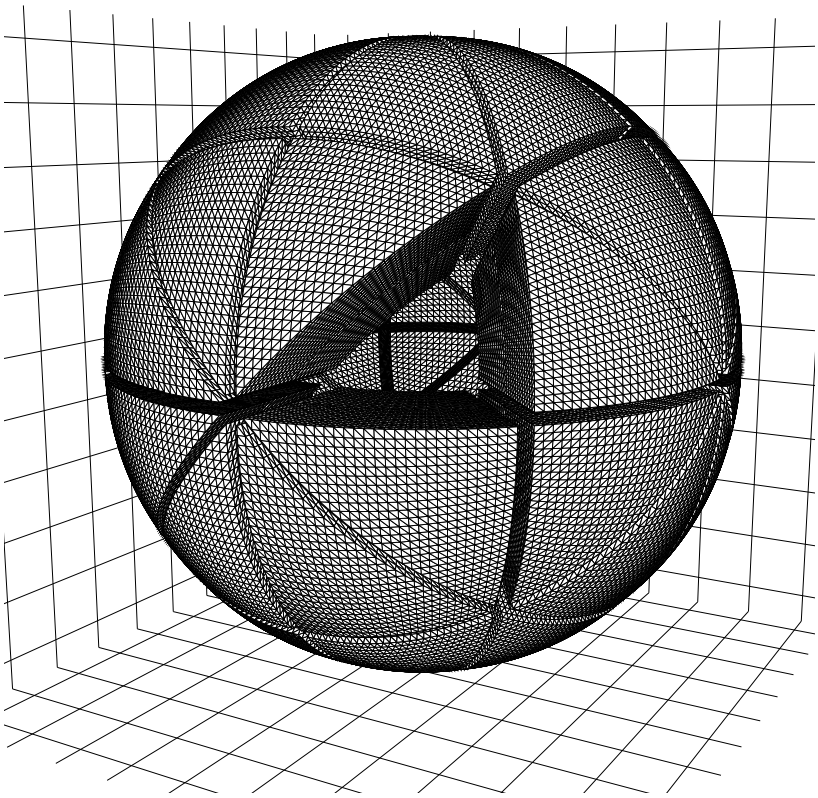

Figure 2.4. A domain decomposition of a spherical shell into 48 subdomains.

Since $A^{(0)}$, $M_1^{(0)}$, $D^{(0)}$, and $M_0^{(0)}$ are symmetric matrices, the numbers of their nonzero components to be stored are as follows,

$$
\begin{aligned}
n_{A^{(0)}} &:= \sum_{1\le i\le n_X^{(0)}} \#\{j\in\Lambda_X^{(0)}\,;\,[A^{(0)}]_{i\,j}\neq 0,\ j\le i\}\,,\\
n_{B^{(0)}} &:= \sum_{1\le i\le n_S^{(0)}} \#\{j\in\Lambda_X^{(0)}\,;\,[B^{(0)}]_{i\,j}\neq 0\}\,,\\
n_{M_1^{(0)}} &:= \sum_{1\le i\le n_X^{(0)}} \#\{j\in\Lambda_X^{(0)}\,;\,[M_1^{(0)}]_{i\,j}\neq 0,\ j\le i\}\,,\\
n_{D^{(0)}} &:= \sum_{1\le i\le n_S^{(0)}} \#\{j\in\Lambda_S^{(0)}\,;\,[D^{(0)}]_{i\,j}\neq 0,\ j\le i\}\,,\\
n_{M_0^{(0)}} &:= \sum_{1\le i\le n_S^{(0)}} \#\{j\in\Lambda_S^{(0)}\,;\,[M_0^{(0)}]_{i\,j}\neq 0,\ j\le i\}\,.
\end{aligned}
$$

We note that $n_{M^{(0)}} = n_{D_0^{(0)}}$, and $n_{M_1^{(0)}} = 3n_{M_0^{(0)}}$. In order to practice Algorithm 2.4, we need integer vectors to realize the bijections $f^{(j)}$, $f_X^{(j)}$ and to store sings $\{\epsilon_k^{(j)}\}_{1\le k\le 3}$ for each j. Therefore, we need the following amounts of memory in bytes for $p(\ge 2)$:

$$
\begin{aligned}
m_A(p) &:= 12n_{A^{(0)}} + 4(n_X^{(0)}+1) + 4p(n_X^{(0)}+3), && (2.22a)\\
m_B(p) &:= 12n_{B^{(0)}} + 4(n_S^{(0)}+1) + 4p(n_S^{(0)}+n_X^{(0)}+3), && (2.22b)\\
m_{M_1}(p) &:= 12n_{M_1^{(0)}} + 4(n_X^{(0)}+1) + 4p\,n_X^{(0)}, && (2.22c)\\
m_D(p) &:= 12n_{D^{(0)}} + 4(n_S^{(0)}+1) + 4p\,n_S^{(0)}, && (2.22d)\\
m_{M_0}(p) &:= 12n_{M^{(0)}} + 4(n_S^{(0)}+1) + 4p\,n_S^{(0)}. && (2.22e)
\end{aligned}
$$

For $p = 1$, they are

$$
\begin{aligned}
m_A(p) &:= 12n_A + 4(n_X+1), && (2.23a)\\
m_B(p) &:= 12n_B + 4(n_S+1), && (2.23b)\\
m_{M_1}(p) &:= 12n_{M_1} + 4(n_X+1), && (2.23c)\\
m_D(p) &:= 12n_D + 4(n_S+1), && (2.23d)\\
m_{M_0}(p) &:= 12n_{M_0} + 4(n_S+1). && (2.23e)
\end{aligned}
$$

We note that $m_{M_0}(p) = m_D(p)$ for $p \ge 1$. We evaluate reduction rate of memory reqiurement for a matrix C by $m_C(p)/m_C(1)$.

Now we employ the P1 element. In Table 2.4 we gave parameters for mesh of tetrahedra of a spherical shell domain. Figure 2.5 shows the number of nodes and the numbers of of nonzero components of the stiffness and mass matrices in the reference subdomain against the number of subdomains. Figure 2.6 shows the

sizes of memroy requirements for stiffness and mass matrices against the number of the subdomains. The third terms of (2.22) show the memory requirements for processes of renumbering indices and changing signs. The sizes of index array $4p\,n_X^{(0)}$ and $4p\,n_S^{(0)}$ increase slightly as the number of subdomains becomes large because there are overlaps of nodes on the interface. Therefore reduction rate of memory requirements does not vary linearly, which are observed most clearly in case of the matrix B in Figure 2.6. With 48 subdomains, reduction rate for the matrix B is 1/17.8.

2.5 Parallel computation

2.5.1 Parallel computation of matrix-vector product

By Algorithm 2.4, a matrix-vector multiplication can be computed with work-vectors in subdomains, and the sub-matrix in the reference subdomain. This algorithm is not only efficient in reducing memory requirements but also in parallel computation. We consider an implementation of Algorithm 2.4 on some parallel computers with shared memory.

We define an interface among the subdomains by

$$\mathcal{F} := \bigcup_{0\le i<j<p} \partial\Omega^{(i)} \cap \partial\Omega^{(j)}\,, \tag{2.24}$$

where $\partial\Omega^{(i)}$ denotes the boundary of $\Omega^{(i)}$. We decompose index set $\Lambda_X^{(i)}$ into a union of node indices in subdomain and that on the interface as follows. We define index sets $\Lambda_{X,\,I}^{(i)}$ and $\Lambda_{X,\,B}^{(i)}$ by

$$\Lambda_{X,\,I}^{(i)} := \{\alpha = [\alpha_0,\,\alpha_1] \in \Lambda_X^{(i)}\,;\, P_{\alpha_0} \in \Omega^{(i)} \cup (\partial\Omega^{(i)} \setminus \mathcal{F}),\ \alpha_1 \in \{1,2,3\}\}\,, \tag{2.25}$$

$$\Lambda_{X,\,B}^{(i)} := \{\alpha = [\alpha_0,\,\alpha_1] \in \Lambda_X^{(i)}\,;\, P_{\alpha_0} \in \partial\Omega^{(i)} \cap \mathcal{F},\ \alpha_1 \in \{1,2,3\}\}\,, \tag{2.26}$$

where P_{α_0} is a node with index α_0. We set

$$\Lambda_{X,\,I} := \bigcup_{0\le i<p} \Lambda_{X,\,I}^{(i)}\,, \tag{2.27}$$

$$\Lambda_{X,\,B} := \bigcup_{0\le i<p} \Lambda_{X,\,B}^{(i)}\,. \tag{2.28}$$

We have

$$I(\alpha) = \{i\} \qquad \text{for } \alpha \in \Lambda_{X,\,I}^{(i)}\,,$$
$$\#I(\alpha) > 1 \qquad \text{for } \alpha \in \Lambda_{X,\,B}^{(i)}\,.$$

Then, Step 3 of Algorithm 2.4 is divided into two procedures:

$$[A\,\vec{u}]_{\alpha^{(j)}} := \epsilon_{k^{(j)}}^{(j)} [\vec{w}^{(j)}]_{\tilde{\alpha}^{(j)}} \quad (\alpha^{(j)} = [\alpha_0^{(j)},\,k^{(j)}] \in \Lambda_{X,\,I}^{(j)}\,,\ j = 0,\ldots,p-1)\,, \tag{2.29}$$

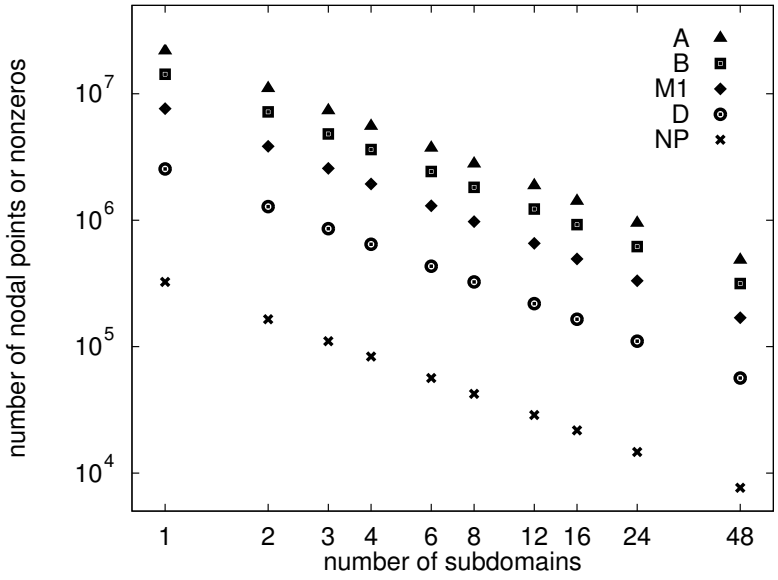

Figure 2.5. Numbers of nonzero components of matrices and nodes in the reference subdomain in case of the spherical shell domain (`A`, `B`, `M1`, `D` denote the numbers of nonzero components of the stiffness and mass matrices $n_{A^{(0)}}$, $n_{B^{(0)}}$, $n_{M_1^{(0)}}$, and $n_{D^{(0)}}$, respectively, and `N` denotes the node number $n_G^{(0)}$.)

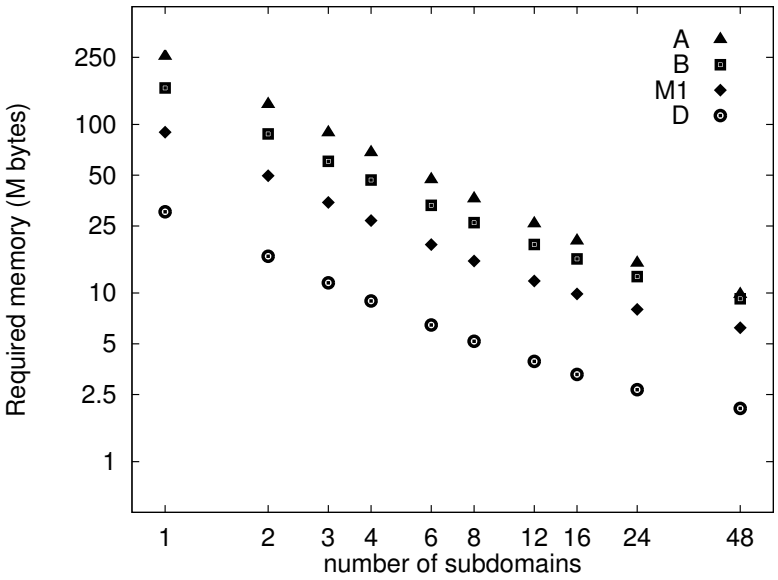

Figure 2.6. Memory requirements to store matrices in case of the spherical shell domain (`A`, `B`, `M1`, `D` denote the sizes of memory requirements $m_A(p)$, $m_B(p)$, $m_{M_1}(p)$, $m_D(p)$, respectively.)

and

$$[A\,\vec{u}]_\alpha := \sum_{j \in I(\alpha)} \epsilon_k^{(j)} [\vec{w}^{(j)}]_{\tilde{\alpha}^{(j)}} \quad (\alpha = [\alpha_0,\, k] \in \Lambda_{X,\,B}). \tag{2.30}$$

Procedure (2.29) are executable independently among subdomains. For parallel execution of (2.30), we prepare a decomposition of the index set $\Lambda_{X,\,B}$,

$$\Lambda_{X,\,B} = \bigcup_{0 \le i \le P-1} \Lambda_{X,\,B}^{(i),P}, \quad \Lambda_{X,\,B}^{(i),P} \cap \Lambda_{X,\,B}^{(j),P} = \emptyset \quad (0 \le i < j < P-1)\,,$$

where P is a positive integer. We note that

$$\Lambda_{X,\,B} = \bigcup_{0 \le i \le p-1} \Lambda_{X,\,B}^{(i)}\,,$$

while indices $\{\Lambda_{X,\,B}^{(i)}\}_{0 \le i < p-1}$ are overlapping.

Now we present a parallel version of Algorithm 2.4. We suppose that the number of processors P is a divisor of subdomain number p. We put $m := p/P$.

Algorithm 2.5

```
parallel do r = 0, ..., P − 1
    do s = 0, ..., m − 1
        j := r × m + s;
        [v^(j)]_δ := ε_n^(j) [u]_{f_X^(j)(δ)}   (δ = [δ_0, n] ∈ Λ_X^(0));
        w^(j) := A^(0) v^(j);
        [A u]_{α^(j)} := ε_{k^(j)}^(j) [w^(j)]_{α̃^(j)}   (α^(j) ∈ Λ_{X,I}^(j));
    enddo
enddo
barrier
parallel do r = 0, ..., P − 1
    [A u]_α := Σ_{j∈I(α)} ε_k^(j) [w^(j)]_{α̃^(j)}   (α = [α_0, k] ∈ Λ_{X,B}^{(r),P});
enddo.
```

In Algorithm 2.5 "barrier" stands for a synchronization among parallel processes. Every process will wait until completion of other processes. Since we use parallel computers with shared memory, there is no requirement for data transfer, which is, however, critical operation in parallel computers with distributed memory.

Remark 2.4 Since we take number P to be a divisor of p, Algorithm 2.5 has almost perfect load balance.

2.5.2 Parallel efficiency on shared-memory computer

We have implemented Algorithm 2.5 on two kinds of parallel computers with shared memory using OpenMP parallel library [33]. We show results of parallel computation on Fujitsu GP7000F and Compaq GS320 in Computing and Communication Center, Kyushu University. Both computers are equipped with 32 processors.

Let p be a processor number and let $t(p)$ be elapsed time with p processors. We evaluate speed-up ratio in parallel execution by $t(1)/t(p)$ and parallel efficiency by $t(1)/(p\,t(p))$, respectively.

In Figures 2.7 and 2.8 we give computation time for the matrix-vector multiplication of a stiffness matrix in the Stokes problem by a vector corresponding to both velocity and pressure unknowns. Stiffness matrix $\mathcal{A}$ is generated by using the finite element mesh in Figure 2.4 with parameter listed in Table 2.4. Length of the vector is $n_X + n_S = 1,298,128$. In the spherical shell domain, the maximum number of subdomains is 48 from Proposition 2.4. In this computation, the domain is divided into the union of 48 subdomains and numbers of processors is taken as divisors of 48 less than 32, i.e. 1, 2, 3, 4, 6, 8, 12, 16, or 24. Figure 2.7 shows that Algorithm 2.5 has very good parallel efficiency. In case of GP7000F, with 24 processors, speed-up ratio is 18.1 and parallel efficiency is 75.4%.

2.5.3 Speed-up on a single processor

Matrix-vector multiplication with CRS format

We have employed the CRS format for sparse matrices to reduce memory requirements. The CRS format is not suitable for random memory access, and hence, it is not good at computation. We will see performance of a multiplication of a matrix by a vector using the CRS format on the workstations and present a strategy for fast computation. Figure 2.9 shows a data structure of the CRS format in the C language[30]. Figure 2.10 shows a program fragment in C to calculate

$$\vec{w} := A\vec{v},$$

where $\vec{v}, \vec{w} \in \mathbb{R}^{n_X}$ and $A \in \mathbb{R}^{n_X \times n_X}$. Here A is symmetric and has n_A nonzero entries. In a multiplication of a matrix by a vector, indirect memory access is required to read and to write components of $\vec{v}$ (`v[ ]`) and $\vec{w}$ (`w[ ]`). Table 2.5 shows elapsed time to complete a matrix-vector multiplication and obtained Mflops in the computation. Here, the size of the vector is $n_X = 973,596$ and the size of nonzero components of the matrix $n_A = 21,909,498$. In total 88.612 million floating point operations have been executed.

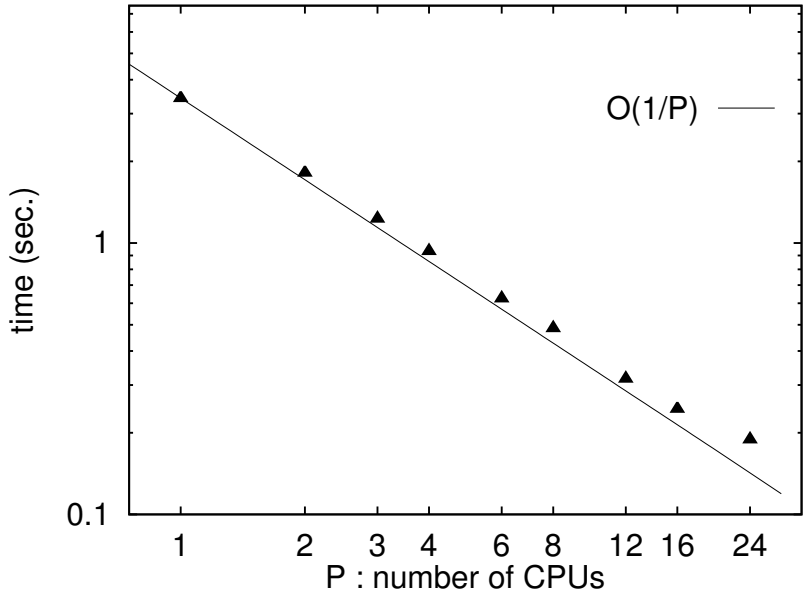

Figure 2.7. Elapsed time in a matrix-vector multiplication by GP7000F

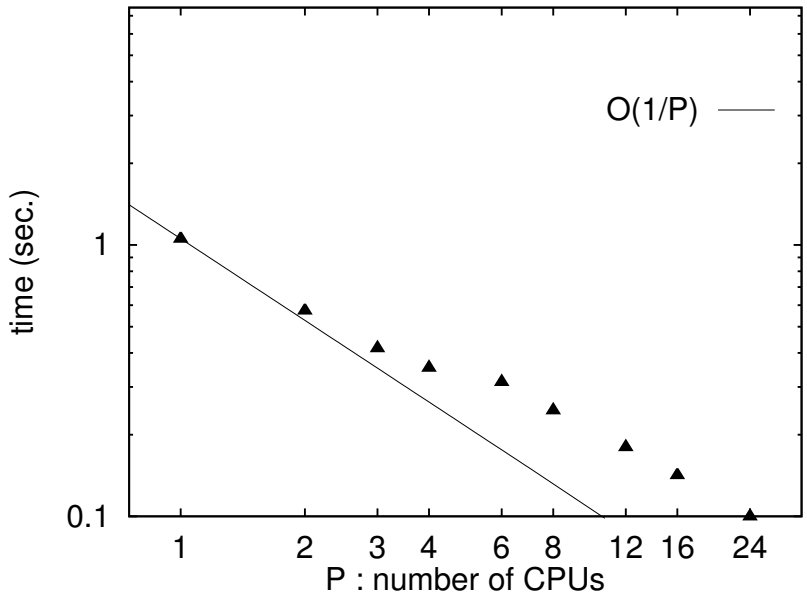

Figure 2.8. Elapsed time in a matrix-vector multiplication by GS320

```
#define SIZE       /****/
#define ROWS1      (SIZE + 1)
#define NONZEROS   /****/

typedef struct {
  int row[ROWS1];
  int index[NONZEROS];
  double value[NONZEROS];
} sparse_matrix;
```

Figure 2.9. Data structure of the CRS format in the C language (`SIZE` denotes the size of the vector n_X, and `NONZEROS` the number of nonzero components of the matrix n_A.)

```
sparse_matrix aa;
double *v, *w;

for (i = 0; i < SIZE; i++) {
  w_tmp = 0.0;
  j_begin = aa.row[i];
  j_end = aa.row[i + 1] - 1;
  for (j = j_begin; j < j_end; j++) {
    j_tmp = aa.index[j];
    w_tmp += aa.value[j] * v[j_tmp];
    w[j_tmp] += aa.value[j] * v[i];
  }
  j_tmp = aa.index[j_end];
  w_tmp += aa.value[j_end] * v[j_tmp];
  w[i] += w_tmp;
}
```

Figure 2.10. A matrix-vector multiplication with the CRS format in the C language

Table 2.5. Elapsed time and flops of a matrix-vector multiplication

computer	elapsed time (sec.)	Mflops
GP7000F	2.028	43.7
GS320	0.446	198.7

Speed-up on a single processor with cache memory

To utilize a cache memory for speed-up of a multiplication of a matrix by a vector, a blocking algorithm [1] is proposed for dense matrices. This algorithm is obtained by applying stripe-mine-and-interchange to an inner loop of a matrix multiplication. This algorithm exploits contiguous memory access in computation of a dense matrix. However, a sparse matrix requires random memory access. Because we need to calculate multiplications of a sparse matrix by vectors, we have to develop a new algorithm.

In Algorithm 2.5, we need to calculate matrix-vector multiplications in m times. To show simply an idea to utilize a cache memory in the processor, we set P to be 1, which means serial computation. In Figure 2.11 we show a program fragment in C to realize the procedure

`do` $j = 0, \ldots, p-1$
$\quad \vec{w}^{(j)} := A^{(0)}\, \vec{v}^{(j)}$;
`enddo` .

The outmost loop is driven by the index `q` over subdomains. We exploit repeat of memory accesses to `aa.index[ ]` and `aa.value[ ]` in the loop on the index `q` to utilize a cache memory in the processor. We call the loop on the index `q` '`q`-*loop*'. The program fragment can be rewritten with interchanges of the `i`-loop or the `j`-loop and the `q`-loop. In Figure 2.12, the `q`-loops over subdomains are located innermostly. We call the program fragment in Figure 2.11 '*loop1*' and one in Figure 2.12 '*loop2*'. In *loop2*, the value of stiffness matrix `aa_value` is stored in a cache memory and accessed m times during execution of the `q`-loop. Therefore, we can get speed-up in *loop2* compared to *loop1*. Table 2.6 shows elapsed times of *loop1* and *loop2*, and speed-up ratios with GP7000F and GS320, where $p = 48$, $n_X^{(0)} = 22,932$ and $n_A^{(0)} = 484,533$.

Table 2.6. Elapsed time of matrix-vector multiplications with *loop1* and *loop2*

computer	elapsed time (sec.)		speed-up
	loop1	*loop2*	
GP7000F	2.013	1.060	1.98
GS320	0.528	0.273	1.93

```
for (q = 0; q < SBDMN; q++) {
  for (i = 0; i < SIZE1; i++) {
    w_tmp = 0.0;
    j_begin = aa.row[i];
    j_end = aa.row[i + 1] - 1;
    for (j = j_begin; j < j_end; j++) {
      j_tmp = aa.index[j];
      w_tmp += aa.value[j] * v[q][j_tmp];
      w[q][j_tmp] += aa.value[j] * v[q][i];
    }
    j_tmp = aa.index[j_end];
    w_tmp += aa.value[j_end] * v[q][j_tmp];
    w[q][i] += w_tmp;
  }
}
```

Figure 2.11. Matrix-vector multiplications with congruent subdomains: *loop1* (`SBDMN` denotes the number of subdomains p, and `SIZE1` the number of index $\#\Lambda_X^{(i)}$.)

Speed-up on a single vector processor

Matrix-vector multiplications with the CRS format in Figure 2.11 are not suitable for vector processing. Because of sparseness of the matrix, the innermost loop is too short to utilize vector registers. In addition, we have another difficulty on vectorization of preconditioned CG methods. A most common preconditioner using incomplete Cholesky decomposition [24] is not easily vectorized especially with the matrix derived from finite element method with an unstructured mesh. There is another kind of storage format for the sparse matrix, the JDS format [3]. The JDS format is constructed from the CRS format by sorting the rows of the matrix in decreasing order of nonzero elements and then extracting element from each row of sorted data in the column-direction. Since length of the innermost loop in matrix-vector multiplications with the JDS format is very long, the JDS format is suitable for vector processing. However, the JDS format is not applicable to the

```
for (i = 0; i < SIZE1; i++) {
  for (q = 0; q < SBDMN; q++) {
    w_tmp[q] = 0.0;
  }
  j_begin = aa.row[i];
  j_end = aa.row[i + 1] - 1;
  for (j = j_begin; j < j_end; j++) {
    j_tmp = aa.index[j];
    aa_value = aa.value[j];
    for (q = 0; q < SBDMN; q++) {
      w_tmp[q] += aa_value * v[j_tmp][q];
      w[j_tmp][q] += aa_value * v[i][q];
    }
  }
  j_tmp = aa.index[j_end];
  for (q = 0; q < SBDMN; q++) {
    w_tmp[q] += aa.value[j_end] * v[j_tmp][q];
    w[i][q] += w_tmp[q];
  }
}
```

Figure 2.12. Matrix-vector multiplications with congruent subdomains: *loop2*

Table 2.7. Vector performances of *loop1* and *loop2*

loop	elapsed time (sec.)	Mflops	average vector length	ratio of vector operation
loop1	2.572	36.2	21.1	94.14
loop2	0.216	431.6	367.6	99.81

preconditioning operation with incomplete Cholesky decomposition. To vectorize the *PCG Algorithm* we exploit the loop that is obtained by domain decomposition. This kind of technique for vectorization is found in [42]. By this strategy the length of the innermost loop becomes p times large as *loop1*, where p is a number of subdomains. In this approach, there is no difficulty to vectorize the procedure of solving linear systems with lower and upper tridiagonal matrices in operation of preconditioning matrix to the residual vectors. We show performance of vector processing on NEC SX-5 in Cybermedia Center, Osaka University. Table 2.7 shows elapsed time, flops, average vector length, and ratio of vector operations obtained in the matrix-vector multiplications. Data of degrees of freedom are the same as in Table 2.6. We observe that in *loop2* the average vector length becomes longer and ratio of vector operations becomes bigger than in *loop1*.

2.6 Summary

We have developed an economical solver with a simple structure for the Stokes problem, which is convenient to implement. Especially, our solver is suitable for large-scale computation.

We have derived a finite element equation expressed with a stiffness matrix on the whole finite element nodes and the orthogonal projection onto the solution space considering boundary conditions and linear constraints to eliminate rigid body rotations and pressure lifting.

We used the CG method with preconditioners that consist of indefinite symmetric matrices. The property of finite termination of the PCG method for a linear system with an indefinite symmetric matrix was verified theoretically, and the convergence property was observed by numerical experiments.

We have proposed an algorithm by domain decomposition into congruent subdomains to reduce the memory requirements. We have obtained simple representations of matrices that refer to sub-matrices in a reference domain. The algorithm presented has three advantages: (i) efficient reduction of the memory requirements, (ii) almost optimal load balance in parallel computing, and (iii) efficient use of a cache memory and of vector registers. We observed these advantages in numerical experiments using an example of stiffness matrix obtained by P1 element with a tetrahedral mesh of a spherical shell domain.

In Chapter 3, we will present the performance of the parallel Stokes solver applied to the problem of Earth's mantle convection, where temperature-dependent viscosity is considered and a more complicated mesh subdivision is employed.

Chapter 3

Application to Earth's mantle convection problem

3.1 Introduction

In this chapter, we review a problem of the Earth's mantle convection and we show the performance of our parallel Stokes solver applied to this problem.

A problem of the Earth's mantle convection is described by the Rayleigh–Bénard equations with an infinite Prandtl number. In geophysical models, some complex rheology is considered. The viscosity of the mantle is strongly dependent on the temperature. The corresponding mathematical model becomes a nonlinear system of the Stokes equations and the convection-diffusion equation in a spherical shell domain. These two equations are coupled by the viscosity, the buoyancy and the convection terms. Many authors have performed numerical simulations to analyze the phenomenon using the spectral method [7], the finite element method [6], and the finite volume method [29, 37, 38]. In former two simulations the viscosity is assumed to be constant, not temperature-dependent. To the best of our knowledge, however, there is no literature discussing the convergence of the numerical solutions. We use the finite element scheme, whose convergence property has been mathematically justified [46, 47]. The domain of the Earth's mantle convection is a spherical shell domain; nevertheless, we can use Cartesian coordinates. This coordinate system is free from singularities on the poles that appear in spherical coordinates [29, 38]. In the finite element method, we can use a non-uniform mesh to capture the drastic change of unknown variables in the boundary layers. There are two difficulties in solving the finite element problem. The first is in solving a problem in a 3-D spherical shell domain, for which large-scale computation is required. The second difficulty is in treatment of temperature-dependent viscosity. For example, with a linearized Arrhenius law [37, 38], the viscosity depends exponentially on the temperature, which demands a robust Stokes solver. We developed a finite element scheme for the isoviscosity [46] and performed numerical simulations [43], then we extended the scheme to temperature-dependent viscosity [47]. In the simulation code, almost all

the time is consumed in solving the Stokes problem. Consequently, it is important to develop a fast Stokes solver. We present an extension of the parallel Stokes solver developed in Chapter 2 to variable viscosity.

Organization of this chapter

In Section 3.2, we review a mathematical model of Earth's mantle convection with temperature-dependent viscosity in a 3-D spherical shell domain. In Section 3.3, we present a finite element scheme for the Rayleigh–Bénard equations with an infinite Prandtl number using piecewise linear elements and stabilization techniques. We use the same finite element spaces for the velocity and the pressure as described in Section 1.5 to treat the Stokes problems with slip boundary conditions. In Section 3.4, we show the P1/P1 stabilized finite element method of penalty type for the Stokes problem with variable viscosity. We present an error estimate of the finite element solution. In Section 3.5, we review an error estimate of the total finite element scheme for the Rayleigh–Bénard equations with an infinite Prandtl number. In Section 3.6, we show finite element equations with stiffness and mass matrices of the Stokes problem and the convection-diffusion problem in the framework of Section 1.3. We present a strategy to reduce memory requirements in computation of the Stokes problem with temperature-dependent viscosity using domain decomposition into congruent subdomains. In Section 3.7, we give numerical results on the evolution problems of Earth's mantle convection phenomena with isoviscosity or temperature-dependent viscosity. Finally, we observe the convergence of the PCG method in the Stokes solver with temperature-dependent viscosity and give the parallel performance of the solver.

3.2 Mathematical model of Earth's mantle convection problem

In this section we review a mathematical model for the Earth's mantle convection. The mantle is considered to be an incompressible fluid in the spherical domain between the Earth surface and its core surface. The core surface is hot and the surface of the Earth is cold. Gravity directs to the center of the Earth. We suppose that the Earth's mantle convection is governed by the Rayleigh–Bénard equations

$$\begin{aligned} \rho\left\{\frac{\partial u}{\partial t} + (u\cdot\nabla)u\right\} + \nabla p - \nabla\cdot[2\mu D(u)] &= -\rho g e^{(r)}, \\ \nabla\cdot u &= 0, \\ \frac{\partial\theta}{\partial t} + u\cdot\nabla\theta - \nabla\cdot(\kappa\nabla\theta) &= f, \end{aligned}$$

where u is the velocity, p the pressure, θ the temperature, ρ the density, μ the viscosity, g the gravity acceleration, $e^{(r)}$ the unit radial vector ($e^{(r)}(x) = x/|x|$), κ the thermal diffusivity, and f the heat source.

The density ρ is, in general, a function of the temperature and the pressure. We, however, introduce the Boussinesq approximation to ρ, i.e., ρ of the buoyancy term is replaced by

$$\rho = \rho_0\{1 - \alpha(\theta - \theta_0)\}\,,$$

where α is the thermal expansion coefficient, and ρ_0 and θ_0 are a representative density and a representative temperature, respectively, and ρ of the inertia term is replaced by ρ_0. We note that the pressure p and a representative pressure p_0 do not appear in the buoyancy term. Besides, we assume that the others, g, κ, and α are positive constants. In a model of rheology of the mantle, the viscosity μ is strongly dependent on the temperature. Ratcliff et al. [37] pointed out the importance of this complex rheology in the spherical shell domain with numerical experiments. So, we treat μ as a function of the temperature θ in a mathematical model.

Through scaling x, t, u, p, θ, f, and μ by the magnitude d, d^2/κ, κ/d, $\mu_0\kappa/\rho_0\, d^2$, $\Delta\theta$, κ/d^2, and μ_0, and translating θ by θ_0 and p by p_0, we obtain non-dimensional equations

$$\begin{aligned}
\frac{1}{Pr}\left\{\frac{\partial u}{\partial t} + (u\cdot\nabla)u\right\} + \nabla p - \nabla\cdot[2\mu(\theta)D(u)] &= Ra\theta e^{(r)},\\
\nabla\cdot u &= 0,\\
\frac{\partial\theta}{\partial t} + u\cdot\nabla\theta - \nabla^2\theta &= f,
\end{aligned}$$

where d is the depth of mantle, $\Delta\theta$ is the difference of temperature, μ_0 is the representative viscosity, and Pr and Ra are Prandtl and Rayleigh numbers defined by

$$Pr := \frac{\mu_0}{\kappa\rho_0}, \quad Ra := \frac{\rho_0 g\alpha\Delta\theta d^3}{\kappa\mu_0}\,,$$

respectively. Since Pr is of order $10^{23} \sim 10^{24}$ in the mantle convection, we omit the inertia term. Scaling again t by $1/Ra$ and u, p, and f by Ra, we obtain

$$-\nabla\cdot[2\mu(\theta)D(u)] + \nabla p = \theta e^{(r)}, \tag{3.1a}$$

$$\nabla\cdot u = 0, \tag{3.1b}$$

$$\frac{\partial\theta}{\partial t} + u\cdot\nabla\theta - \frac{1}{Ra}\nabla^2\theta = f\,. \tag{3.1c}$$

We assume that the viscosity $\mu = \mu(\theta)$ is a positive function of θ.

Equations (3.1) are considered in a spherical shell domain

$$\Omega := \{x \in \mathbb{R}^3\,;\, 0 < R_1 < |x| < R_2\}\,.$$

In case of the Earth $R_1 = 11/9$ and $R_2 = 20/9$. Let Γ_1 and Γ_2 be inner and outer boundaries and Γ be the whole boundary $\Gamma_1 \cup \Gamma_2$. The slip boundary conditions for

u and the Dirichlet boundary conditions for θ are imposed on Γ :

$$u \cdot n = 0\,, \tag{3.2a}$$
$$D(u)n \times n = 0\,, \tag{3.2b}$$
$$\theta = \theta_\Gamma\,, \tag{3.2c}$$

where n is the unit outer normal and θ_Γ is a given temperature on the boundary. Initial condition for θ at $t = 0$,

$$\theta = \theta^0 \tag{3.3}$$

completes a mathematical model of the Earth's mantle convection, where θ^0 is a given temperature.

3.3 P1/P1/P1 finite element scheme for the Rayleigh–Bénard equations with an infinite Prandtl number

Here we present a finite element approximation to the problem (3.1), (3.2), and (3.3). We employ a cheap element combination P1/P1/P1, that is, velocity, pressure, and temperature are all approximated by piecewise linear elements. Considering again computation cost, we employ the stabilization of the penalty type for the Stokes problem with temperature-dependent viscosity. Details of this method are described in Section 3.4. Since Ra is high in our problem, the convection term is dominant in $(3.1c)$. It is well known that the classical Galerkin method to the convection-dominant equation causes numerical instability. To solve the equation without numerical instability under high Rayleigh numbers, we use the stream upwind Petrov/Galerkin (SUPG) method [14, 21]. We note that, in case of P1 element, the GLS method [28, 21] to convection-diffusion problem $(3.1c)$ with P1 element is equivalent to the SUPG method.

Let Ω_h be a polyhedral approximation to Ω and $\mathcal{T}_h$ be a partition of $\bar{\Omega}_h$ by tetrahedra, where h is the maximum diameter of tetrahedral elements. The boundary of Ω_h is denoted by Γ_h. We consider a regular family of subdivisions $\{\mathcal{T}_h\}$, $h \downarrow 0$, satisfying the inverse assumption [15], i.e., there exists a positive constant τ such that $\min_{K\in\mathcal{T}_h} h_K \geq \tau h$ for all $h > 0$. We use the same finite element spaces W_h, V_h, Q_h as defined for the Stokes problem with slip boundary conditions in Section 1.5. We introduce a finite element affine space $\Psi_h(\theta_\Gamma)$ and a finite element space Ψ_h defined by

$$\Psi_h(\theta_\Gamma) := \{\psi_h \in S_h \ ; \ \psi_h(P) = \theta_\Gamma(P) \ (\forall P)\}\,,$$
$$\Psi_h := \Psi_h(0)\,,$$

where P stands for a node on Γ_h.

We prepare the following bilinear and trilinear forms: for $u,\ v \in X_h,\ q \in S_h$, and $\theta,\ \psi \in S_h$,

$$
\begin{aligned}
a_h(\mu\,;\,u,v) &:= 2\int_{\Omega_h} \mu D(u) : D(v)\,dx,\\
b_h(v,q) &:= -\int_{\Omega_h} \nabla\cdot v\,q\,dx,\\
d_h(p,q) &:= \sum_{K\in\mathcal{T}_h} h_K^2(\nabla p,\nabla q\,;\,K),\\
c_{0\,h}(\theta,\psi) &:= \frac{1}{Ra}\int_{\Omega_h} \nabla\theta\cdot\nabla\psi\,dx,\\
c_{1\,h}(u,\theta,\psi) &:= \frac{1}{2}\left\{\int_{\Omega_h}(u\cdot\nabla)\theta\,\psi\,dx - \int_{\Omega_h}(u\cdot\nabla)\psi\,\theta\,dx\right\}.
\end{aligned}
$$

Let Δt be a time increment. The total numbers of time steps $[T/\Delta t]$ is denoted by N_T. The value of v_h at $t = n\Delta t$, $n \in [0, N_T]$ is denoted by v_h^n. Let Π_h be an interpolation operator from $L^2(\Omega)$ to S_h.

We approximate the time derivative $\partial\theta/\partial t$ at $t = (n+1)\Delta t$ by the difference quotient $D_{\Delta t}\theta^n := (\theta^{n+1} - \theta^n)/\Delta t$. A stabilized finite element method for (3.1), (3.2), and (3.3) is to find $\{(u_h^n,\, p_h^n,\, \theta_h^n)\}_{n=0}^{N_T} \subset V_h \times Q_h \times \Psi_h(\theta_\Gamma)$ satisfying

$$
a_h(\Pi_h\mu(\theta_h^n)\,;\,u_h^n,\,v_h) + b_h(v_h,\,p_h^n) = (\theta_h^n e_h^{(r)},\,v_h)_h\,, \tag{3.4a}
$$

$$
b_h(u_h^n,\,q_h) - \delta d_h(p_h^n, q_h) = 0\,, \tag{3.4b}
$$

$$
\begin{aligned}
&(D_{\Delta t}\theta_h^n,\ \psi_h)_h + c_{0\,h}(\theta_h^{n+1},\ \psi_h) + c_{1\,h}(u_h^n,\,\theta_h^{n+1},\,\psi_h)\\
&+ \sum_{K\in\mathcal{T}_h} \tau_K(D_{\Delta t}\theta_h^n + u_h^n\cdot\nabla\theta_h^{n+1},\ u_h^n\cdot\nabla\psi_h\,;\,K)\\
&= (f_h^{n+1},\,\psi_h)_h + \sum_{K\in\mathcal{T}_h}\tau_K(f_h^{n+1},\,u_h^n\cdot\nabla\psi_h\,;\,K)\,,
\end{aligned} \tag{3.4c}
$$

for all $(v_h, q_h, \psi_h) \in V_h \times Q_h \times \Psi_h$ and $n \in [0, N_T]$ with an initial condition

$$
\theta_h^0 = \Pi_h\theta^0\,.
$$

A positive constant δ is a stability parameter for the Stokes equations and τ_K is also a stability parameter for the convection-diffusion equation defined by

$$
\tau_K := \min\left\{\frac{\Delta t}{2}, Ra\frac{h_K^2}{12}, \frac{h_K}{2U_K}\right\},
$$

where h_K is the diameter of element K, $U_K = |u_h(G_K)|$, and G_K is the barycenter of K.

When θ_h^n is given, a unique solution (u_h^n, p_h^n) of (3.4a) and (3.4b) is obtained, and then with u_h^n, a unique solution θ_h^{n+1} of (3.4c) is obtained. Hence, starting from the initial value θ_h^0, we can obtain the finite element solution $\{(u_h^n, p_h^n, \theta_h^n)\}$.

3.4 Stokes problem with variable viscosity and stabilized finite element approximation

We again assume that Ω be a polyhedral bounded domain for simplicity. We consider the linear Stokes problem with variable viscosity to find u and p satisfying

$$-2\nabla\cdot[\mu(x)D(u)]+\nabla p=f \quad \text{in } \Omega\,, \tag{3.5a}$$

$$\nabla\cdot u=0 \quad \text{in } \Omega\,, \tag{3.5b}$$

with Dirichlet boundary conditions $u=g$ on Γ, $\int_\Gamma g\cdot n ds=0$. Here $\mu\in C^1(\bar\Omega)$ means the viscosity coefficient. We suppose that there exists a positive number μ_0 such that

$$\mu(x)\geq\mu_0 \quad (x\in\bar\Omega)\,. \tag{3.6}$$

We define a bilinear form $A_\delta^{\text{Penalty}}(\cdot\,;\,\cdot)$ on $X_h\times S_h$, which depends on the viscosity, in the same manner of Section 1.5 by

$$\begin{aligned}A_\delta^{\text{Penalty}}(\mu_h\,;\,u_h,\,p_h\,;\,v_h,\,q_h) &:= a(\mu_h\,;\,u_h,v_h)+b(v_h,p_h)+b(u_h,q_h)\\ &\quad -\delta d(p_h,q_h)\,,\end{aligned} \tag{3.7}$$

where δ is a positive constant and μ_h is an interpolant of μ by P1 element. We recall the functional on $X_h\times S_h$ defined by

$$F_\delta^{\text{Penalty}}(v_h,\,q_h):=(f,v_h)\,. \tag{1.39}$$

Since we consider the problem in the polyhedral bounded domain with Dirichlet boundary conditions, we use the affine space for the velocity $V_h(g)$ defined in (1.30) and the function space for the pressure Q_h defined in (1.32). Now we present a finite element problem by a penalty-type stabilization. The problem is to find $(u_h,p_h)\in V_h(g)\times Q_h$ satisfying

$$A_\delta^{\text{Penalty}}(\mu_h\,;\,u_h,\,p_h\,;\,v_h,\,q_h)=F_\delta^{\text{Penalty}}(v_h,q_h) \tag{3.8}$$

for all $(v_h,q_h)\in V_h\times Q_h$.

Remark 3.1 We can also consider a finite element approximation with a GLS-type stabilization technique to (3.5) by using a bilinear form on $X_h\times S_h$,

$$\begin{aligned}A_\delta^{\text{GLS}}(\mu_h\,;\,u_h,\,p_h\,;\,v_h,\,q_h) &:= a(\mu_h\,;\,u_h,v_h)+b(v_h,p_h)+b(u_h,q_h)\\ &\quad -\delta\sum_{K\in\mathcal{T}_h}h_K^2(-2\,\nabla\cdot[\mu_h D(u_h)]+\nabla p_h,-2\,\nabla\cdot[\mu_h D(v_h)]+\nabla q_h\,;\,K)\,,\end{aligned}$$

and a functional on $X_h\times S_h$,

$$F_\delta^{\text{GLS}}(v_h,q_h):=(f,\,v_h)-\delta\sum_{K\in\mathcal{T}_h}h_K^2(f,\,-2\,\nabla\cdot[\mu_h D(v_h)]+\nabla q_h\,;\,K)\,.$$

The bilinear form $A_\delta^{\rm GLS}$ includes the term $\nabla\cdot[\mu_h D(u_h)]$. It does not vanish elementwise for the P1 element unless μ is constant, which increases considerably the computation cost in the 3-D problem. That is the reason why we use the penalty-type stabilization.

Lemma 3.1 *There exists a positive constant c_0 such that for all positive number h and all $v_h \in V_h$*

$$a(\mu_h\,;\,v_h, v_h) \geq \mu_0 c_0 ||v_h||_1^2\,.$$

A proof is obtained by the positiveness of the viscosity coefficient (3.6).

Proposition 3.1 *We have*

$$A_\delta^{\rm Penalty}(\mu_h\,;\,v_h,\,q_h\,;\,v_h,\,-q_h) \geq \mu_0 c_0 ||v_h||_1^2 + \delta |q_h|_h^2\,,$$

for all $v_h \in V_h$ and $q_h \in Q_h$.

From Lemma 3.1 and Proposition 3.1, we have the following stability inequality on $A_\delta^{\rm Penalty}(\cdot\,;\,\cdot)$ as same as the constant viscous case in Section 1.5, Lemma 1.12.

Lemma 3.2 *There exists a positive constant α_1 such that for all positive h it holds that*

$$\inf_{(u_h,p_h)\in V_h\times Q_h}\sup_{(v_h,q_h)\in V_h\times Q_h}\frac{A_\delta^{\rm Penalty}(u_h,\,p_h\,;\,v_h,\,q_h)}{||(u_h,p_h)||_{V\times Q}||(v_h,q_h)||_{V\times Q}} \geq \alpha_1\,.$$

Lemma 3.3 *Let $(u,p) \in H^2(\Omega)^3 \times H^1(\Omega)$ be a solution of the continuous problem (3.5) and $(u_h,p_h) \in V_h(g)\times Q_h$ be a solution of the finite element equation (3.8). There exist constants c_1 and c_2 such that for all positive number h and $(v_h,q_h) \in V_h \times Q_h$*

$$|A_\delta^{\rm Penalty}(\mu_h\,;\,u-u_h,\,p-p_h\,;\,v_h,\,q_h)| \leq c_1\,h\,|\mu|_{1,\infty}||u||_1||v_h||_1 + c_2 h||p||_1||q_h||_0.$$

Here, $|\mu|_{1,\infty} := \max_j \sup_{x\in\Omega}|\partial_j\mu(x)|$.
Proof. We have

$$\begin{aligned}
&\Big|\ A_\delta^{\rm Penalty}(\mu_h\,;\,u-u_h,\,p-p_h\,;\,v_h,\,q_h)\ \Big| \\
&\quad= \Big|\ 2\int_\Omega \mu_h(D(u)-D(u_h)) : D(v_h)\,dx + b(v_h,p-p_h) + b(u-u_h,q_h) \\
&\qquad - \delta d(p-p_h,q_h)\ \Big| \\
&\quad= \Big|\Big\{\ 2\int_\Omega \mu D(u) : D(v_h)\,dx + b(v_h,p) + b(u,q_h)\Big\} - \delta d(p,q_h) \\
&\qquad - \Big\{\ 2\int_\Omega \mu_h D(u_h) : D(v_h)\,dx + b(v_h,p_h) + b(u_h,q_h) - \delta d(p_h,q_h)\ \Big\} \\
&\qquad - 2\int_\Omega (\mu-\mu_h) D(u) : D(v_h)\,dx\ \Big| \\
&\quad\leq \delta|d(p,q_h)| + 2\sup_{x\in\Omega}|\mu(x)-\mu_h(x)|||D(u)||_0\,||D(v_h)||_0\,.
\end{aligned}$$

The assertion follows from the inequality (1.36) and an estimate

$$\sup_{x\in\Omega} |\mu(x) - \mu_h(x)| \le c\, h\, |\mu|_{1,\infty}\,,$$

which is obtained from the fact that μ_h is the P1-interpolant of μ. □

Using Lemmas 3.2 and 3.3, and Theorem 1.2 (i), we obtain the existence and uniqueness of the finite element solution and its error estimate for the variable viscous case in the same manner of the Theorem 1.4 (ii) for the constant viscous case.

Theorem 3.1
(i) *There exists a unique solution* $(u_h, p_h) \in V_h(g) \times Q_h$ *of* (3.8) *for all positive* δ.
(ii) *Let* $(u, p) \in H^2(\Omega)^3 \times H^1(\Omega)$ *be the solution of the problem* (3.5). *Let the unique solution* $(u_h, p_h) \in V_h(g) \times Q_h$ *of the problem* (3.8). *There exists a positive constant* c *such that for all positive* h

$$||u - u_h||_1 + ||p - p_h||_0 \le c\, h\{||u||_2 + ||p||_1\}\,.$$

We can consider the linear Stokes equations (3.5) with variable viscosity and slip boundary conditions in a spherical shell domain. The function space $\tilde{V}_h$ defined in (1.34) is used for the velocity, and $\tilde{Q}_h$ defined in (1.35) is used for the pressure. Convergence of the finite element solution by the penalty-type stabilization is also obtained.

Theorem 3.2
(i) *There exists a positive constant* h_0 *such that for all* $h \in (0, h_0]$, *problem* (3.8) *in* $\tilde{V}_h \times \tilde{Q}_h$ *has a unique solution* $(u_h, p_h) \in \tilde{V}_h \times \tilde{Q}_h$.
(ii) *Let* $(u, p) \in H^2(\Omega)^3 \times H^1(\Omega)$ *be the solution of the Stokes problem with slip boundary conditions. There exist positive constants* c *and* h_0 *such that for all* $h \in (0, h_0]$,

$$||u - u_h||_1 + ||p - p_h||_0 \le c\, h\{||u||_2 + ||p||_1\}\,.$$

3.5 Error estimate of the scheme for the Rayleigh–Bénard equations with an infinite Prandtl number

We will review an error estimate of the finite element scheme for the Rayleigh–Bénard equations with an infinite Prandtl number. It is known that the scheme (3.4) is unconditionally stable and that the finite element solutions converge to the exact one [47]. To describe the error estimate, we prepare a norm. Let X be a Hilbert space. We consider a sequence $v_h = \{v_h^n\}_{n=0}^{N_T} \subset X$. We define $\ell^q(X)$-norm

for the sequence,

$$||v_h||_{\ell^q(X)} := \begin{cases} \left\{ \Delta t \sum_{n=0}^{N_T} ||v_h^n||_X^q \right\}^{1/q} & \text{for } \ 1 \le q < \infty \,, \\ \\ \max\limits_{0 \le n \le N_T} ||v_h^n||_X & \text{for } \ q = \infty \,. \end{cases}$$

Theorem 3.3 *Let* (u, p, θ) *be a solution of* (3.1), (3.2), *and* (3.3) *such that*

$$\begin{aligned} u \in & C([0,T]\,; H^2(\Omega)^3) \cap H^1(0,T\,; H^1(\Omega)^3), \\ p \in & C([0,T]\,; H^1(\Omega)), \\ \theta \in & H^1(0,T\,; H^2(\Omega)) \cap H^2(0,T\,; L^2(\Omega)). \end{aligned}$$

Suppose that the initial value θ_h^0 *satisfies*

$$||\theta_h^0 - \theta^0||_0 \le c\, h ||\theta^0||_1 \ .$$

Then there exist positive constants $c_* = c_*(T; u, p, \theta)$ *and* h_0 *such that for all* $\Delta t > 0$ *and* $h \in (0, h_0]$,

$$\begin{aligned} ||\theta_h - \theta||_{\ell^\infty(L^2)} \,, \quad & ||\frac{1}{\sqrt{Ra}}(\theta_h - \theta)||_{\ell^2(H^1)} \,, \\ ||\sqrt{\tau}\, u_h \cdot \nabla(\theta_h - \theta)||_{\ell^2(L^2)} \,, \quad & ||\sqrt{\tau}\,(D_{\Delta t} + u_h \cdot \nabla)(\theta_h - \theta)||_{\ell^2(L^2)} \,, \\ ||u_h - u||_{\ell^\infty(H^1)} \,, \ \textit{and} \ \ & ||p_h - p||_{\ell^\infty(L^2)} \quad \le c_*\,(\Delta t + h) \,. \end{aligned}$$

Here, in the third and the forth terms, the values of the sequence at the $(n+1)$*th step are evaluated as*

$$\begin{aligned} & \sqrt{\tau}\ u_h^n \cdot \nabla \psi_h^{n+1} \,, \\ & \sqrt{\tau}\ \left\{ (\psi_h^{n+1} - \psi_h^n)/\Delta t + u_h^n \cdot \nabla \psi_h^{n+1} \right\} \,, \end{aligned}$$

where $\psi_h^n := \theta_h^n - \theta(\cdot, n\Delta t)$. *A weighted norm is defined by*

$$||\sqrt{\tau}\,\phi_h||_{L^2(\Omega)} = \left\{ \sum_{K \in \mathcal{T}_h} \tau_K ||\phi_h||_{L^2(K)}^2 \right\}^{1/2} \ .$$

A proof in the case of isoviscosity is found in [46]. Using the error estimate for the Stokes problem with variable viscosity in Theorem 3.2, we obtain the desired result in case of temperature-dependent viscosity.

3.6 Finite element equations in matrix form

We will represent the finite element equations of (3.4) in a matrix form. Let $A(\vec{\theta}^n)$ be a stiffness matrix defined by

$$[A(\vec{\theta}^n)]_{\alpha\,\beta} := 2\int_{\Omega_h} \Pi_h \mu(\theta_h^n) D(\varphi_\beta) \,:\, D(\varphi_\alpha) dx \quad (\alpha, \beta \in \Lambda_X)\,. \tag{3.9}$$

Other stiffness matrices and mass matrices for the Stokes problem, B, M_1, D, M_0 are the same as $(2.3b)$, $(2.3c)$, $(2.3d)$, and $(2.3e)$. We define stiffness matrices for the convection-diffusion equation by

$$\begin{aligned}
[C_0]_{\mu\,\nu} &:= c_{0\,h}(\psi_\nu, \psi_\mu) && (\mu, \nu \in \Lambda_M)\,,\\
[C_1(\vec{u}^n)]_{\mu\,\nu} &:= c_{1\,h}(u_h^n, \psi_\nu, \psi_\mu) && (\mu, \nu \in \Lambda_M)\,,\\
[D_1(\vec{u}^n)]_{\mu\,\nu} &:= \sum_{K\in\mathcal{T}_h} \tau_K(\psi_\nu, u_h^n \cdot \nabla\psi_\mu\,;\, K) && (\mu, \nu \in \Lambda_M)\,,\\
[D_2(\vec{u}^n)]_{\mu\,\nu} &:= \sum_{K\in\mathcal{T}_h} \tau_K(u_h^n \cdot \nabla\psi_\nu, u_h^n \cdot \nabla\psi_\mu\,;\, K) && (\mu, \nu \in \Lambda_M)\,.
\end{aligned}$$

We recall the solution spaces for the Stokes equations,

$$\begin{aligned}
\vec{V} &:= \{\vec{v} \in \mathbb{R}^{n_X} \;;\; (\vec{v},\, \vec{n}^{(j)})_{\mathbb{R}^{n_X}} = 0 \;(j \in \Lambda_\Gamma),\; (M_1\vec{v},\, \vec{v}^{(i)})_{\mathbb{R}^{n_X}} = 0 \;(i \in \{1,2,3\})\},\\
\vec{Q} &:= \{\vec{q} \in \mathbb{R}^{n_S} \;;\; (M_0\vec{q},\, \vec{1})_{\mathbb{R}^{n_S}} = 0\}\,,
\end{aligned}$$

where Λ_Γ is the index set of the nodes of the boundary Γ, and $\vec{n}^{(j)}$ and $\vec{v}^{(i)}$ are defined in (2.4) and (2.5), respectively. We define an affine space $\vec{\Psi}(\theta_\Gamma)$ and a vector space $\vec{\Psi}$ by

$$\begin{aligned}
\vec{\Psi}(\theta_\Gamma) &:= \{\vec{\psi} \in \mathbb{R}^{n_S} \;;\; (\vec{\psi},\, \vec{e}^{(j)})_{\mathbb{R}^{n_S}} = \theta_\Gamma(P_j) \;(j \in \Lambda_\Gamma)\}\,,\\
\vec{\Psi} &:= \vec{\Psi}(0)\,.
\end{aligned}$$

The finite element scheme (3.4) can be rewritten as the problem to find $(\vec{u}^n,\, \vec{p}^{\,n},\, \vec{\theta}^{n+1}) \in \vec{V} \times \vec{Q} \times \vec{\Psi}(\theta_\Gamma)$ for n $(0 \le n < N_T)$ satisfying,

$$(A(\vec{\theta}^n)\vec{u}^n + B^T\vec{p}^{\,n},\, \vec{v})_{\mathbb{R}^{n_X}} = (\vec{g}(\vec{\theta}^n),\, \vec{v})_{\mathbb{R}^{n_X}} \quad \text{for all } \vec{v} \in \vec{V}\,, \tag{3.10a}$$

$$(B\vec{u}^n - \delta D\vec{p}^{\,n},\, \vec{q})_{\mathbb{R}^{n_S}} = 0 \quad \text{for all } \vec{q} \in \vec{Q}\,, \tag{3.10b}$$

$$\begin{aligned}
(\{\, \frac{1}{\Delta t}M_0 + C_0 + C_1(\vec{u}^n) + \frac{1}{\Delta t}D_1(\vec{u}^n) + D_2(\vec{u}^n) \,\}\, \vec{\theta}^{n+1}\,, \vec{\psi})_{\mathbb{R}^{n_S}} =\\
(\{\, \frac{1}{\Delta t}M_0 + \frac{1}{\Delta t}D_1(\vec{u}^n) \,\}\, \vec{\theta}^{n}\,, \vec{\psi})_{\mathbb{R}^{n_S}} + (M_0\vec{f}^{n+1} + D_1(\vec{u}^n)\vec{f}^{n+1},\, \vec{\psi})_{\mathbb{R}^{n_S}}\\
\text{for all } \vec{\psi} \in \vec{\Psi}\,,
\end{aligned} \tag{3.10c}$$

with initial temperature $\vec{\theta}^0$. Here $\vec{g}(\vec{\theta}^n)$ corresponds to the buoyancy term $\theta_h^n e_h^{(r)}$, and $\vec{f}^{n+1}$ to the heat source term f_h^{n+1}.

The discretized Stokes problem (3.10a), (3.10b) is solved by the *PCG Algorithm* (the CG method with an indefinite preconditioning matrix, described in Section 2.3). To obtain $\vec{\theta}^{n+1}$ as the solution of (3.10c), we use the GCR method [19].

In the each subdomain the stiffness matrix is defined by

$$[A^{(i)}(\vec{\theta}^n)]_{\alpha\,\beta} := 2\int_{\Omega_h^{(i)}} \Pi_h\mu(\theta_h^n)D(\varphi_\beta)\ :\ D(\varphi_\alpha)dx \quad (\alpha,\beta\in\Lambda_X^{(i)})\,.$$

Remark 3.2 Since the viscosity function is interpolated by P1 element and φ_α is a base function of P1 element, we have

$$\begin{aligned}[A^{(i)}(\vec{\theta}^n)]_{\alpha\,\beta} &= \sum_{K\in\mathcal{T}_h^{(i)}} 2\int_K \Pi_h\mu(\theta_h^n)D(\varphi_\beta) : D(\varphi_\alpha)dx \\ &= \sum_{K\in\mathcal{T}_h^{(i)}} \frac{1}{4}\{\sum_{1\le j\le 4}\mu(\theta_h^n(P_j))\}2D(\varphi_\beta) : D(\varphi_\alpha)\int_K dx\,.\end{aligned}$$

Here, $\{P_j\}_{1\le j\le 4}$ stand for the nodes of the element K. We note that $\frac{1}{4}\sum_{1\le j\le 4}\mu(\theta_h^n(P_j))$ is equal to $\Pi_h(\mu(\theta_h))$ at the barycenter of K. We recall that an element stiffness matrix on K in case of isoviscosity is

$$\{2D(\varphi_\beta) : D(\varphi_\alpha)\mathrm{vol}(K)\}_{\alpha\,\beta}\quad(\alpha,\beta\in\Lambda_K)\,,$$

where Λ_K is an index set defined by

$$\Lambda_K := \{\alpha=[\alpha_0,\,\alpha_1]\in\Lambda_X\,;\,\{P_{\alpha_0}\}\text{ are nodes of }K,\ \alpha_1\in\{1,2,3\}\}\,.$$

Total stiffness matrix can be generated through multiplying the element stiffness matrix by mean value of the viscosity over every element. Therefore, keeping element stiffness matrices we can reduce the computation cost in generating stiffness matrix at each time step.

Remark 3.3 Since the viscosity varies in the domain, the stiffness matrix in ith subdomain $A^{(i)}$ cannot be transformed from $A^{(0)}$ in the reference subdomain with any modification of signs corresponding to the orthogonal transform $R^{(i)}$. We can slightly reduce memory requirements with the CRS format even in case of temperature-dependent viscosity. Under Assumption 2.2 nodes in subdomains are transformed by the orthogonal transformation from ones in the reference subdomain, and then relative positions of neighbor nodes do not change. Therefore, positions of non-zero element of the stiffness matrix in subdomains are common up to renumbering indices. In case of isoviscosity, the size of memory requirements without domain decomposition $m_A(1) = 12n_A + 4(n_X+1)$ is reduced to $m_A(p) = 12n_{A^{(0)}} + 4(n_X^{(0)}+1) + 4p(n_X^{(0)}+3)$. In case of temperature-dependent

viscosity it is reduced to $m_A(p) = 8n_A + 4n_{A^{(0)}} + 4(n_X^{(0)} + 1) + 4p(n_X^{(0)} + 3)$. Figure 3.1 shows the sizes of memory requirements for stiffness matrix A in both cases of isoviscosity and temperature-dependent viscosity. With 48 subdomains, reduction rates for the matrix A, which is defined in Section 2.4.3, are 1/25.9 and 1/1.47 for the isoviscous case and the temperature-dependent viscous case, respectively. Since other stiffness matrices and mass matrices are the same as ones in the isoviscous case, memory requirements for them are reduced.

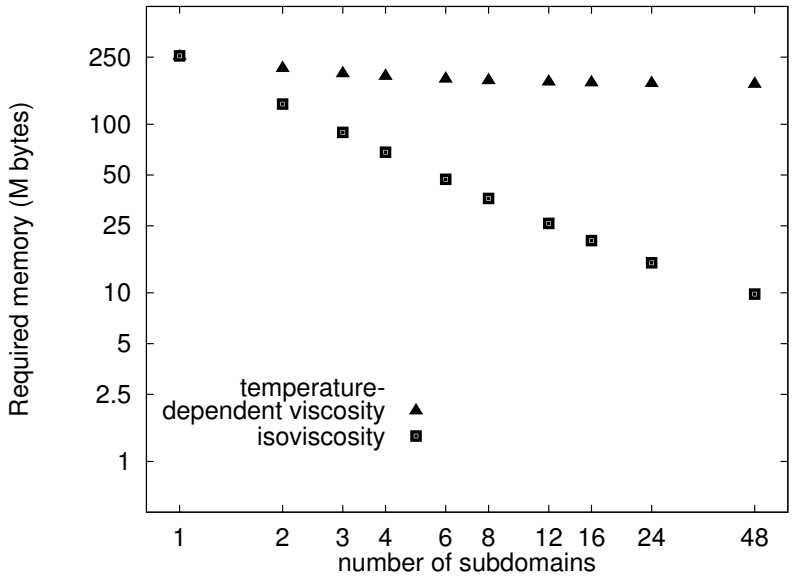

Figure 3.1. Required memories to store stiffness matrix A in isoviscous and temperature-dependent viscous cases

3.7 Numerical results on Earth's mantle convection

3.7.1 Simulation of evolution problem of Earth's mantle convection

We present numerical results of the problem (3.1), (3.2), and (3.3). The temperature θ_Γ on the boundary is given by

$$\theta_\Gamma = 1 \ \text{ on } \Gamma_1, \quad \theta_\Gamma = 0 \ \text{ on } \Gamma_2.$$

We use a linearized Arrhenius law for the viscosity [37, 38],

$$\mu(\theta) = \exp[-(\theta - 1/2)\log b].$$

Here, b is a positive number describing the ratio of the maximum to minimum viscosity. The viscosity μ is normalized at $\theta = 1/2$. The heat source $f = 0$ and the initial temperature θ^0 is given by

$$\theta^0(r, \varphi, \psi) = \theta^*(r) + \epsilon \sin \pi(\frac{R_2 - r}{R_2 - R_1}) Y_3^2(\varphi, \psi)\,, \tag{3.11}$$

where (r, φ, ψ) are spherical coordinates, $\theta^*(r)$ is the conduction profile without convection defined by

$$\theta^*(r) = \frac{R_1}{R_2 - R_1}(\frac{R_2}{r} - 1)\,,$$

Y_n^m is the normalized spherical harmonic function of degree n and order m, and $\epsilon = 0.1$. This initial condition was used in [38]. We set $Ra = 7,000$, which is 10 times as large as the critical Rayleigh number 711.95 in the isoviscous case [50].

We performed numerical simulations for this problem by the stabilized finite element scheme (3.4). Figure 3.2 shows the mesh and Table 3.1 shows discretization parameters for the computation. An eighth part of the shell is cut off to show an interior mesh. We employ a non-uniform mesh, keeping the diameters of the surface triangles of tetrahedra on the inner and outer spheres nearly equal. Elements are densely located near the boundaries to approximate well the temperature in the boundary layers. The stability parameter for the Stokes equations is 0.5×10^{-2}.

Figures 3.3 and 3.4 show evolutions of the isothermal surfaces with $b = 1$ and $b = 100$, starting from the initial temperature (3.11). The isothermal surfaces $\theta_h = 0.2, 0.5$, and 0.8 are drawn from inner to outer of the spherical shell domain. We can observe four 'plumes' in both cases of viscosities. In case of temperature-dependent viscosity, the viscosity increases and the velocity decreases at the place where the temperature is low. The plume heads, therefore, flatten much more than in the isoviscosity case.

Ratcliff et al. [38] showed some temperature distributions at the steady state with various pairs of Rayleigh number and viscosity ratio. To reduce computation cost, they exploited symmetry of the steady solution and reduced the problem in a half of the spherical shell domain assuming reflective boundary conditions on its artificial boundary. Similar tendency of the shape of the plume heads at $\theta = 0.2$ was reported. By our solver, we can obtain solutions of evolution problems of the convection in the whole spherical shell domain, and besides, we can observe influences of the temperature-dependent viscosity on the plumes clear than their work.

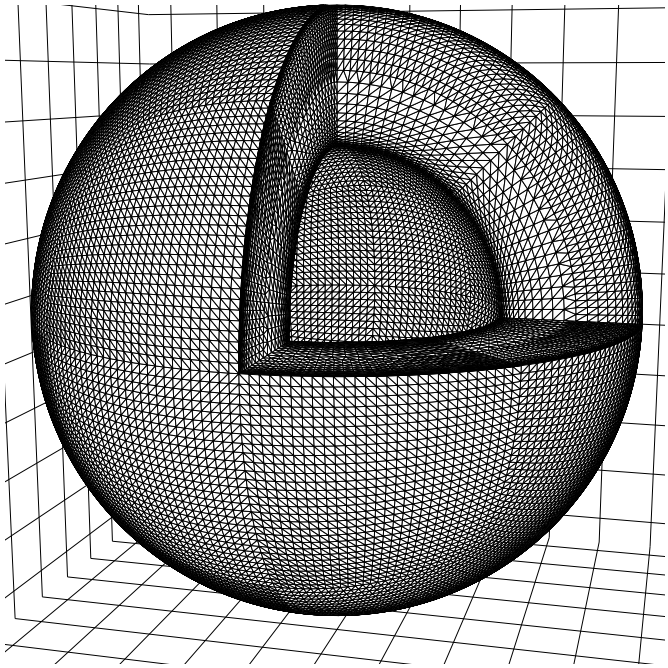

Figure 3.2. Non-uniform mesh concentrated near the boundary

Table 3.1. Discretization parameters (n_G : numbers of nodes, n_E : numbers of elements, n_V : degrees of freedom of velocity, n_Q : degrees of freedom of pressure)

n_G	n_E	n_V	n_Q	$n_V + n_Q$	h	Δt
324,532	1,868,544	948,629	324,531	1,273,160	0.145	2.5

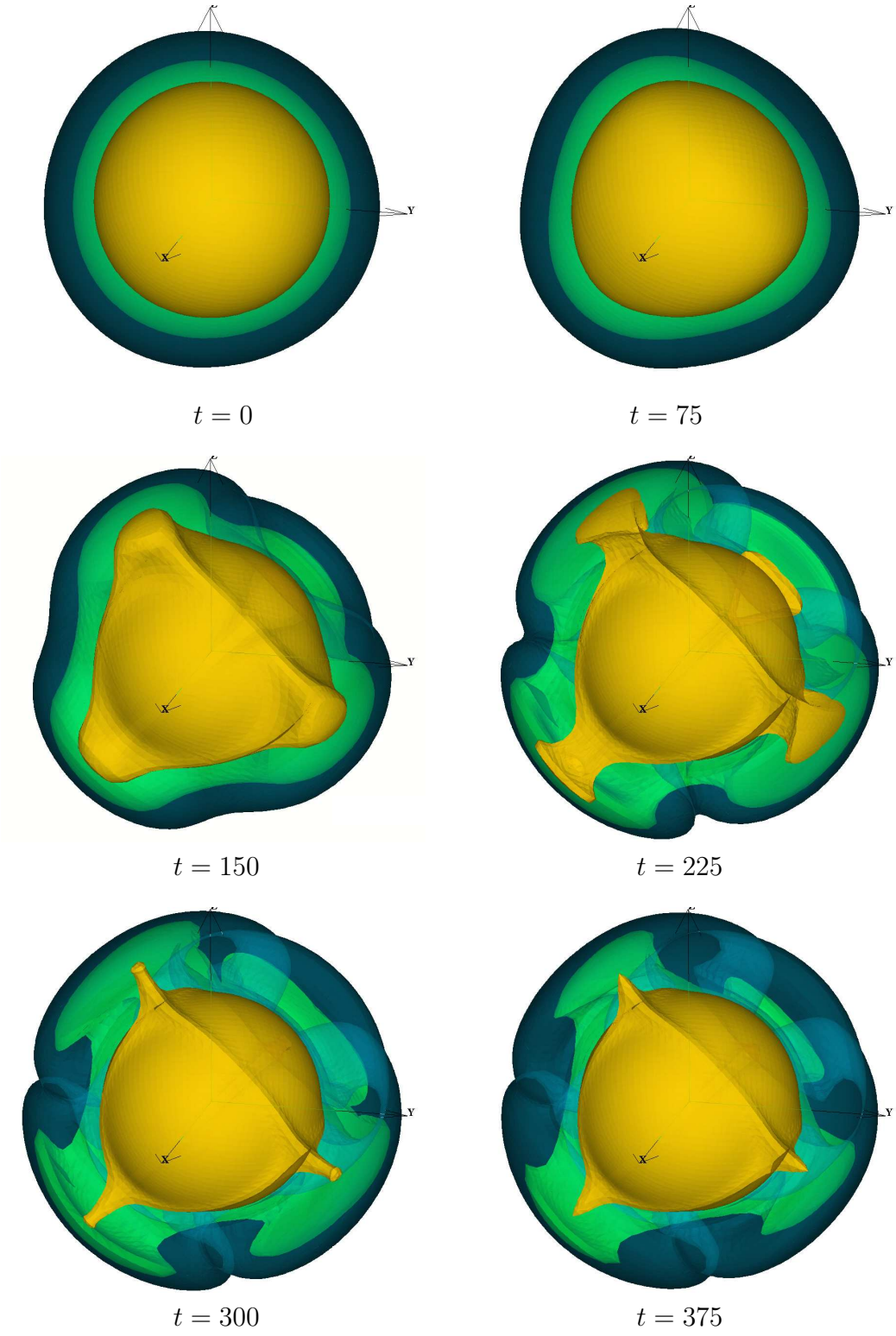

Figure 3.3. Isothermal surfaces of θ_h in the case $b = 1$; $t = 0$, $t = 75$, $t = 150$, $t = 225$, $t = 300$, and $t = 375$

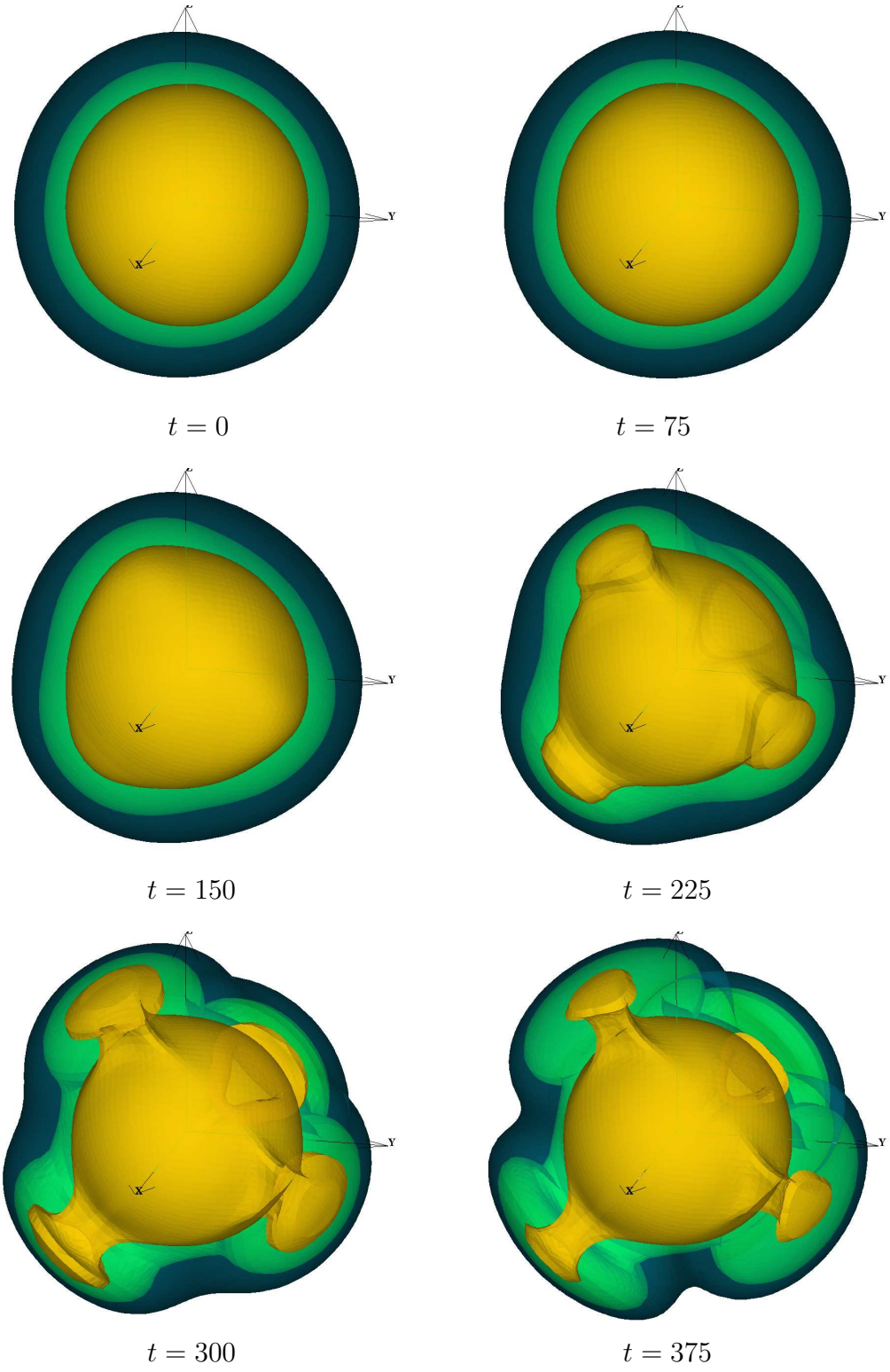

Figure 3.4. Isothermal surfaces of θ_h in the case $b = 100$; $t = 0$, $t = 75$, $t = 150$, $t = 225$, $t = 300$, and $t = 375$

3.7.2 Convergence of the Stokes solver with temperature-dependent viscosity

We demonstrate the convergence of the *PCG Algorithm* to solve the Stokes problem with temperature-dependent viscosity in the Earth's mantle convection. Finite element equations to be solved are $(3.10a)$ and $(3.10b)$ in the solution space with the slip boundary conditions. We prepare a preconditioner defined by

$$Q_3 := \mathcal{P} \begin{pmatrix} \left\{ \begin{pmatrix} \widetilde{A_{II}^{(i)}(\vec{\theta}^n)}^{-1} & 0 \\ 0 & -\frac{1}{\delta}\widetilde{D_{II}^{(0)}}^{-1} \end{pmatrix} \right\}_{0 \le i < p} & 0 \\ 0 & \begin{pmatrix} \widehat{A_{BB}^{(i)}(\vec{\theta}^n)}^{-1} & 0 \\ 0 & -\frac{1}{\delta}\widehat{D_{BB}^{(0)}}^{-1} \end{pmatrix} \end{pmatrix} \mathcal{P}\,.$$

Here $\widetilde{M}$ stands for an incomplete Cholesky decomposition of a matrix M, and $\widehat{M}$ stands for a diagonal matrix with diagonal elements of M. Matrices with subscript II correspond to the index set of inner nodes in a subdomain, e.g., $\Lambda_{X,I}^{(i)}$ defined in (2.25). Matrices with subscript BB correspond to the index set of interface, e.g., $\Lambda_{X,B}$ defined in (2.28).

Figures 3.5 and 3.6 show convergence histories of the *PCG Algorithm* with preconditioner Q_3 for the Stokes problem with $b = 1$ and $b = 100$, respectively. In both cases of viscosity ratio, temperature distribution at $t = 225$ is used. In both cases, the *PCG Algorithm* converges with relative residual 10^{-6} after smaller than 1,600 iterations for the problem with 1,273,160 degrees of freedom. As the viscosity ratio becomes bigger, convergence of *PCG Algorithm* becomes slower slightly. Nevertheless we get the solution after a small number of iterations compared to the degrees of freedom.

3.7.3 Parallel performance of the Stokes solver

Preconditioning operation by matrix Q_3 can be performed in parallel. Figure 3.7 shows parallel performance of the Stokes solver with $b = 100$ using Fujitsu GP7000F. Because the *PCG Algorithm* does not depend on the viscosity ratio, the same parallel performance is obtained in the case of higher viscosity ratio. We can conclude that our Stokes solver using the CG method with congruent domain decomposition has good parallel performance.

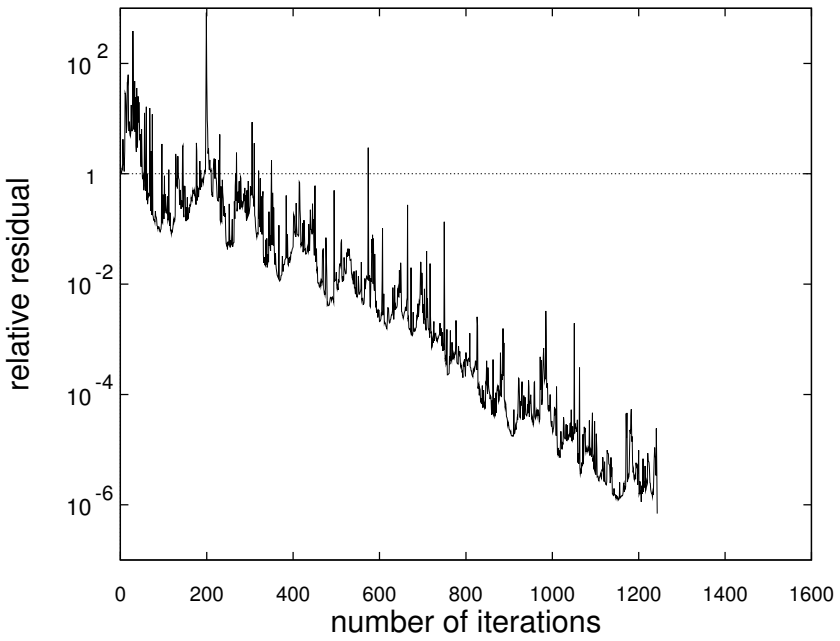

Figure 3.5. Convergence history of the Stokes problem with $b = 1$

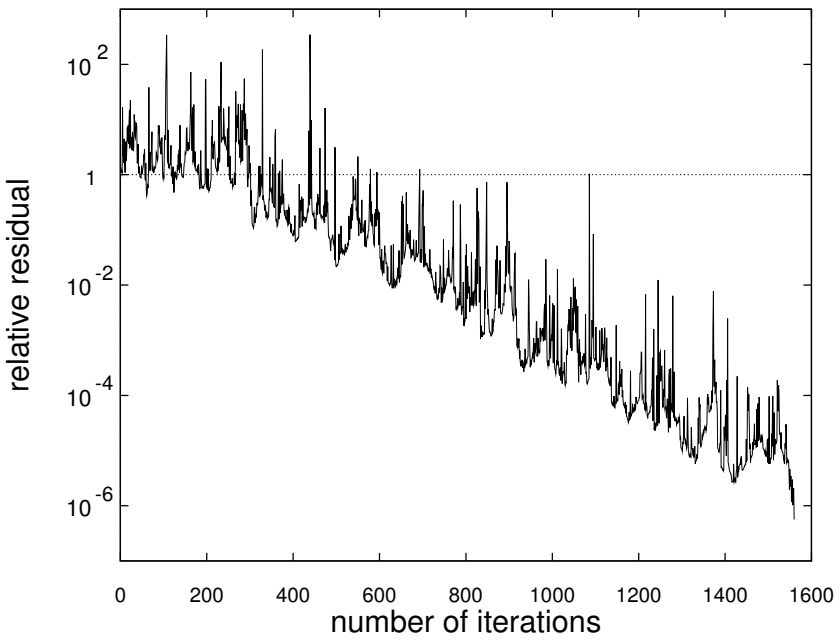

Figure 3.6. Convergence history of the Stokes problem with $b = 100$

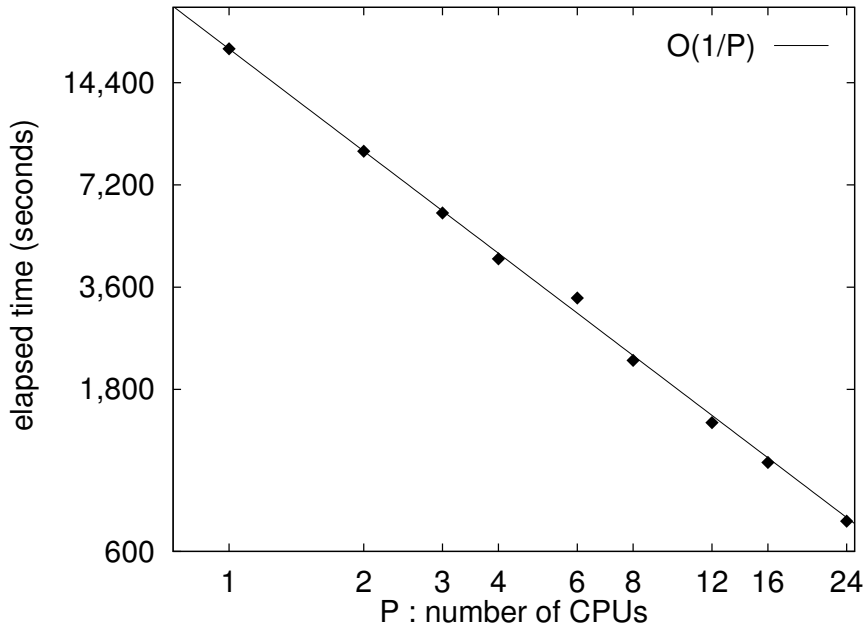

Figure 3.7. Elapsed time in the Stokes solver with $b = 100$ by GP7000F

3.8 Summary

In this chapter, we have shown an application of the parallel Stokes solver to the problem of the Earth's mantle convection. We have reviewed a mathematical model of the Earth's mantle convection that is described by the Rayleigh–Bénard equations with an infinite Prandtl number and temperature-dependent viscosity. The system of equations consists of the Stokes equations and the convection-diffusion equation. We have presented a finite element approximation to the system using stabilization techniques of the penalty type for the Stokes equations and the SUPG type for the convection-diffusion equation.

We have extended the stabilized scheme of the penalty type to the Stokes equations with variable viscosity, and have obtained an error estimate. Because the viscosity depends on the temperature distribution in the domain, the values of the stiffness matrix corresponding to the strain rate tensor are different among the positions of a node in the reference subdomain and of the transformed nodes in the subdomains. Nevertheless, the locations of the nonzero entries of the matrix are common over the subdomains. Each entry of the stiffness matrix is generated from mean values of the viscosity on the elements to which the node belongs, and element stiffness matrices. We could reduce computation cost drastically by storing element stiffness matrices only in the reference subdomain. We concluded that a memory reduction technique using domain decomposition into congruent subdomains is applicable, with some modification, to the temperature-dependent viscosity case.

We have shown the convergence of the PCG method by numerical experiments.

A preconditioner is constructed in block-wise fashion with sub-matrices whose indices correspond to the nodes in the interior of subdomains and to the nodes on the interface among subdomains. We observed the convergence of the PCG solver with relative residual of 10^{-6} after fewer than 1,600 iterations for the problem with more than one million degrees of freedom and a viscosity ratio of 100. This computation required only 13 minutes using the parallel computer, Fujitsu GP7000F, with 24 processors. Therefore, we can say that our parallel Stokes solver has the capability to find a large-scale finite element solution of a 3-D geophysical problem in acceptable computation time.

Conclusion

We have developed a parallel finite element Stokes solver that can perform large-scale 3-D computation, using current computer resources. For its application to the Earth's mantle convection, we considered the linear Stokes equations with variable viscosity in a spherical shell domain with slip boundary conditions imposed. Our solver consists of efficient algorithms to achieve fast computation, with minimal memory requirements. These algorithms are consistent with the framework of the finite element method, and are simple to implement on parallel computers.

In General Introduction to this thesis, we have described the problems that may be caused by some techniques used in conventional finite element codes. We have resolved these problems and developed four new techniques as follows.

(i) We have used the same interpolation for both velocity and pressure by P1 element, which is the most economical element from the viewpoint of memory consumption. Because, as is well known, a combination of P1/P1 elements does not satisfy the inf-sup condition, we have used a stabilization technique. Considering the cost of computation in the case of variable viscosity, we have employed a penalty-type stabilization technique. In the case of constant viscosity, some error estimates for both the penalty-type and GLS-type schemes are known. In this thesis, we have also obtained an error estimate for the penalty-type scheme for the Stokes problems with variable viscosity.

(ii) We have introduced the solution space whose functions satisfy conditions on the normal components of the velocity (a part of the slip boundary conditions), and constraints to eliminate rigid body rotations and pressure lifting. We have written the finite element equation in matrix form using the stiffness matrix, which is constructed by standard finite element base functions independently of the boundary conditions, and an orthogonal projection onto the solution space. We have also expressed the orthogonal projection by inner products of the unknown vector, and vectors corresponding to the unit outer normal on the boundary, the rigid body rotations, and the constant pressure. A finite element solver to find a solution in the Krylov subspace by the CG method has been obtained. The idea of using a combination of the stiffness matrix defined on the all finite element nodes, and the orthogonal projection onto the solution space makes the finite element solver simple and allows memory reduction and parallel computation by the domain decomposition in (iv). Our solver is consistent with the framework of finite element methods and is applicable to finite element equations with general bilinear forms and with

several types of boundary conditions imposed.

(iii) We have verified that the CG method with an indefinite symmetric preconditioning matrix can be applied to linear systems with indefinite symmetric coefficient matrices. We have shown that with successful execution of the preconditioned CG (PCG) method, the linear system can be solved in a Krylov subspace. In addition, we have shown a necessary and sufficient condition for no-breakdown of the PCG method. We have also presented a practical version of the PCG method by adding projection operations to avoid the accumulation of round-off errors.

(iv) We have presented a method to reduce memory requirements for finite element matrices by domain decomposition into congruent subdomains. If a domain has a form of symmetry, it can be decomposed into subdomains that are transformed from the reference subdomain by orthogonal matrices whose components consist of -1, 0, and 1. Using this domain decomposition, the finite element matrices in the subdomains are expressed by a sub-matrix in the reference domain with a slight modification. The modification is performed by renumbering the indices and changing signs corresponding to the orthogonal matrices. We have proposed a strategy to reduce the memory requirements using domain decomposition into congruent subdomains, and have also proposed a procedure to calculate the matrix-vector product, using work-vectors in the subdomains and the sub-matrix in the reference domain. In numerical experiments using an example of a stiffness matrix obtained by P1 element with a tetrahedral mesh of the spherical shell domain and the decomposition of the domain into 48 congruent subdomains, we observed three advantages of our domain decomposition algorithm: (a) efficient reduction of memory requirements, (b) almost optimal load balance in parallel computing, and (c) efficient use of a cache memory and of vector registers. In the case of the Stokes equations with temperature-dependent viscosity, by applying our domain decomposition algorithm to the element stiffness matrix, the calculation cost of the stiffness matrix corresponding to the strain rate tensor was considerably reduced.

We observed the convergence property and parallel performance of our solver to the linear Stokes equations with temperature-dependent viscosity. We have constructed the preconditioner with an incomplete Cholesky decomposition in block-wise fashion and with multiplications of the orthogonal projections from both sides. We have achieved the convergence of the PCG solver with a relative residual 10^{-6} after fewer than 1,600 iterations for a problem with more than one million degrees of freedom and a viscosity ratio of 100. This computation required 13 minutes using the parallel computer, Fujitsu GP7000F, with 24 processors. Using our solver, we have obtained solutions of evolution problems of Earth's mantle convection, and have observed specific influences of the temperature-dependent viscosity on the flow pattern.

We conclude that our parallel Stokes solver has opened a practical way to large-scale simulation of geophysical problems of the Earth's mantle convection phenomena.

Appendix A

A solver of finite element equation in conventional finite element codes and its limitations

A.1 A generalization of a conventional solver

We first recall the problem (1.18) to obtain a solution of the finite element equation (1.10): to find $\vec{u}_0 \in \vec{Y}$ satisfying

$$(\mathcal{A}\vec{u}_0,\, \vec{v}\,)_{\mathbb{R}^{nZ}} = (\vec{f} - \mathcal{A}\vec{u}_g,\, \vec{v}\,)_{\mathbb{R}^{nZ}}\,, \tag{1.18}$$

for a fixed $\vec{u}_g \in \vec{Y}(g)$. Here $\vec{Y}$ and $\vec{Y}(g)$ are the vector space and the affine space defined in (1.13) and (1.14), respectively. $\mathcal{A}$ is the $n_Z \times n_Z$ matrix defined in (1.15) and $\vec{f}$ is the n_Z-vector defined in (1.16).

We will comment a method to find a solution of (1.18) when $\mathcal{A}$ is symmetric. It is a generalization of solvers in conventional finite element codes [5]. In this method, a regular coefficient matrix of the linear system is obtained by fixing some data on finite element nodes and removing some rows and columns of the stiffness matrix. Then, the solution is adjusted by restoring data from the kernel of the stiffness matrix. We rewrite the right-hand term of (1.18). Let $\vec{b} \in \mathbb{R}^{nZ}$. We consider a problem to find $\vec{u} \in \vec{Y}$ satisfying

$$(\mathcal{A}\vec{u},\, \vec{v}\,)_{\mathbb{R}^{nZ}} = (\vec{b},\, \vec{v}\,)_{\mathbb{R}^{nZ}}, \tag{A.1}$$

for all $\vec{v} \in \vec{Y}$. Let $n_Y := \dim \vec{Y}$. $R(\mathcal{A})$ and $N(\mathcal{A})$ denote the image and the kernel of the matrix $\mathcal{A}$, respectively.

We make an assumption on the matrix $\mathcal{A}$.

Assumption A.1 *The matrix $\mathcal{A}$ is symmetric with* $\dim R(\mathcal{A}) = n_Y$.

Let P be an orthogonal projection of $\mathbb{R}^{nZ}$ onto $R(\mathcal{A})$.

We prepare subspaces $\vec{V}_1$ and $\vec{V}_2$ whose dimensions are n_Y, and suppose that subspaces satisfy the following assumption.

Assumption A.2 *Both subspaces $\vec{V}_1$ and $\vec{V}_2$ satisfy*

$$\vec{V}_i \cap N(\mathcal{A}) = \{0\} \qquad (i = 1, 2)\,.$$

Proposition A.1 *Under Assumptions A.1 and A.2, we have*

$$\vec{V}_i^{\perp} \cap R(\mathcal{A}) = \{0\}$$

for $i = 1, 2$.
A proof is easily obtained by expanding $\vec{V}_i^{\perp}$ and $N(\mathcal{A})$ with their bases.

Lemma A.1 *Under Assumptions A.1 and A.2, $\mathcal{A}$ is regular in both $\vec{V}_1$ and $\vec{V}_2$.*
Proof. We will show the injectivity of $\mathcal{A}$ in $\vec{V}_i$. Let $\vec{u} \in \vec{V}_i$ satisfy

$$(\mathcal{A}\vec{u},\, \vec{v}\,)_{\mathbb{R}^{nZ}} = 0$$

for all $\vec{v} \in \vec{V}_i$. We have $\mathcal{A}\vec{u} \in \vec{V}_i^{\perp} \cap R(\mathcal{A})$. By Proposition A.1, we obtain $\mathcal{A}\vec{u} = 0$. Therefore, we have $\vec{u} \in \vec{V}_i \cap N(\mathcal{A})$. Then the injectivity follows from Assumption A.2. The injectivity leads to the surjectivity of $\mathcal{A}$ in $\vec{V}_i$. □

Corollary A.1 *Let $i = 1$ or $i = 2$. Under Assumptions A.1 and A.2, a problem to find $\vec{x}_i \in \vec{V}_i$ satisfying*

$$(\mathcal{A}\vec{x}_i,\, \vec{v}\,)_{\mathbb{R}^{nZ}} = (\vec{b},\, \vec{v}\,)_{\mathbb{R}^{nZ}} \tag{A.2}$$

for all $\vec{v} \in \vec{V}_i$ has a unique solution.

Theorem A.1 *Under Assumptions A.1 and A.2, suppose that $\vec{x}_i \in \vec{V}_i$ be the solution of* (A.2) *for $i = 1, 2$. For each $i = 1, 2$, let $\vec{r}_i \in \vec{V}_i^{\perp}$ satisfy*

$$(I - P)\vec{r}_i = -(I - P)\vec{b}\,. \tag{A.3}$$

Then we have

$$\vec{x}_1 = \vec{x}_2 + \mathcal{A}^{\dagger}(\vec{r}_1 - \vec{r}_2) + \vec{c}, \tag{A.4}$$

where $\vec{c} \in N(\mathcal{A})$.
Proof. Since P is the orthogonal projection onto $R(\mathcal{A})$, we have

$$\begin{aligned}(I - P)(\mathcal{A}\vec{x}_i - \vec{b} - \vec{r}_i) &= (I - P)(\mathcal{A}\vec{x}_i - \left\{P\vec{b} + (I - P)\vec{b}\right\} - \vec{r}_i) \\ &= (I - P)(\mathcal{A}\vec{x}_i - P\vec{b}) - (I - P)^2\vec{b} - (I - P)\vec{r}_i \\ &= 0\,.\end{aligned}$$

Consequently, we obtain

$$\begin{aligned}&(\mathcal{A}\vec{x}_i - \vec{b} - \vec{r}_i,\, \vec{w}\,)_{\mathbb{R}^{nZ}} = 0 \quad \text{for all } \vec{w} \in N(\mathcal{A})\,, \\ &(\mathcal{A}\vec{x}_i - \vec{b} - \vec{r}_i,\, \vec{v}\,)_{\mathbb{R}^{nZ}} = 0 \quad \text{for all } \vec{v} \in \vec{V}_i\,.\end{aligned}$$

Since $N(\mathcal{A}) \cap \vec{V}_i = \{0\}$ and $\dim N(\mathcal{A}) + \dim \vec{V}_i = n_Z$, these two relations lead to

$$\mathcal{A}\vec{x}_i - \vec{b} = \vec{r}_i \,.$$

Then we get

$$\begin{aligned}
\mathcal{A}(\vec{x}_1 - \vec{x}_2) &= (\vec{r}_1 - \vec{r}_2) \\
&= P(\vec{r}_1 - \vec{r}_2) + (I - P)(\vec{r}_1 - \vec{r}_2) \\
&= P(\vec{r}_1 - \vec{r}_2) + (I - P)(-\vec{b} - (-\vec{b})) \\
&= P(\vec{r}_1 - \vec{r}_2) \,.
\end{aligned}$$

Since the last term belongs to $R(\mathcal{A})$, we have

$$\vec{x}_1 = \vec{x}_2 + \mathcal{A}^\dagger(\vec{r}_1 - \vec{r}_2) + \vec{c},$$

where $\vec{c} \in N(\mathcal{A})$. $\square$

Remark A.1 We can determine the vectors $\vec{r}_i$ and $\vec{c}$ in Theorem A.1 as follows. We put $m := n_Z - n_Y$. Let $\{\vec{w}^{\,(1)}, \ldots, \vec{w}^{\,(m)}\}$ be a basis of $N(\mathcal{A})$, and let $\{\vec{v}_i^{\,(1)}, \ldots, \vec{v}_i^{\,(m)}\}$ be a basis of $\vec{V}_i^{\,\perp}$ for $i = 1, 2$, respectively. The vector $\vec{r}_i \in \vec{V}_i^{\,\perp}$ that meets the condition (A.3) is obtained by solving the problem to find $\{\mu_l\}_{l=1}^m$ satisfying

$$\sum_{l=1}^{m} (\vec{v}_i^{\,(l)},\, \vec{w}^{\,(k)})_{\mathbb{R}^{n_Z}} \mu_l = -(\vec{b},\, \vec{w}^{\,(k)})_{\mathbb{R}^{n_Z}} \quad \text{for all } 1 \le k \le m \,.$$

By Assumption A.2, $\{\mu_l\}_{l=1}^m$ are uniquely obtained. Then we have $\vec{r}_i = \sum_{l=1}^m \mu_l \vec{v}_i^{\,(l)}$. Since $\vec{x}_1 \in \vec{V}_1$, the vector $\vec{c}(= \sum_{l=1}^m \nu_l \vec{w}^{\,(l)}) \in N(\mathcal{A})$ is obtained by solving the problem to find $\{\nu_l\}_{l=1}^m$ satisfying

$$\sum_{l=1}^{m} (\vec{w}^{\,(l)},\, \vec{v}_1^{\,(k)})_{\mathbb{R}^{n_Z}} \nu_l = -(\mathcal{A}^\dagger(\vec{r}_1 - \vec{r}_2) + \vec{x}_2,\, \vec{v}_1^{\,(k)})_{\mathbb{R}^{n_Z}} \quad \text{for all } 1 \le k \le m \,. \tag{A.5}$$

Now we present a solver for (A.1), which is a generalized version of solvers used in conventional finite element codes. We make an assumption on right-hand vector $\vec{b}$.

Assumption A.3 *The vector in the right-hand term of* (A.1) *satisfies*

$$\vec{b} \in R(\mathcal{A}) \,.$$

Let $\vec{V}_1$ be the solution space $\vec{Y}$. We prepare $\vec{V}_2$ defined by

$$\vec{V}_2 := \{\vec{v} \in \mathbb{R}^{n_Z} \,;\, (\vec{v},\, \vec{e}^{\,(\mu)})_{\mathbb{R}^{n_Z}} = 0 \quad (\,\mu \in \Lambda_0\,)\},$$

where $\vec{e}^{\,(\mu)}$ is a unit vector and Λ_0 is an index set with $\#\Lambda_0 = n_Z - n_Y$. Here $\#\Lambda$ stands for the total number of elements of a set Λ. We note that Λ_0 should be so chosen as to let $\vec{V}_2$ satisfy Assumption A.2. We put $\Lambda_1 := \Lambda_Z \setminus \Lambda_0$.

Algorithm A.1

Step 1. Let $\tilde{\mathcal{A}}$ be an $n_Y \times n_Y$ matrix defined by

$$[\tilde{\mathcal{A}}]_{\tilde{\mu}\,\tilde{\nu}} = (\mathcal{A}\vec{e}^{(\nu)},\, \vec{e}^{(\mu)})_{\mathbb{R}^{n_Z}} \quad (\mu, \nu \in \Lambda_1)\,.$$

Let $\vec{g}$ be an n_Y-vector defined by

$$[\vec{g}]_{\tilde{\mu}} = (\vec{b},\, \vec{e}^{(\mu)})_{\mathbb{R}^{n_Z}} \quad (\mu \in \Lambda_1)\,.$$

Here, $\tilde{\mu} \in \{1, \ldots, n_Y\}$ stands for an index renumbered from $\mu \in \Lambda_1$. We note that $\tilde{\mathcal{A}}$ is nothing but a reduced matrix of $\mathcal{A}$, where rows and columns with indices in Λ_0 are removed.

Step 2. Find a solution $\vec{y}_2 \in \mathbb{R}^{n_Y}$ satisfying

$$\tilde{\mathcal{A}}\vec{y}_2 = \vec{g}$$

by a linear solver of an iterative method or a direct method.

Step 3. Find a solution $\vec{c} \in \mathbb{R}^{n_Z}$ satisfying (A.5) with $\vec{r}_1 = \vec{r}_2 = 0$.

Step 4. Renumbering indices of the vector $\vec{y}_2$, we get $\vec{x}_2 \in \mathbb{R}^{n_Z}$ as follows,

$$\begin{aligned} [\vec{x}_2]_\mu &= [\vec{y}_2]_{\tilde{\mu}} && (\mu \in \Lambda_1)\,, \\ [\vec{x}_2]_\mu &= 0 && (\mu \in \Lambda_0)\,. \end{aligned}$$

A desired solution of (A.1) is obtained as

$$\vec{u} = \vec{x}_2 + \vec{c}\,.$$

We note that under Assumption A.3, we need not to calculate $A^\dagger(\vec{r}_1 - \vec{r}_2)$ in Step 3.

In many cases of finite element equations, Assumption A.1 on the dimension of the image of the stiffness matrix is satisfied. The condition is that the dimension of the image of the stiffness matrix should be equal to the dimension of the solution space. But, in general, Assumption A.1 is not satisfied. For example, perturbations from discretizatations of the continuous problem cause the inconvenience. In Appendix A.2, we will show an example of finite element equation where Algorithm A.1 is not applicable.

A.2 Limitations of a conventional solver in application to the Stokes problems

We will show Algorithm A.1 in Appendix A.1, which is used in conventional finite element codes, is applicable to the finite element equation of the Stokes problems or not.

Case of Dirichlet boundary conditions

In the *Dirichlet Problem*, it is required to remove pressure lifting. For simplicity, we assume that $g = 0$, i.e., homogeneous Dirichlet data. Let $\Lambda_V := \Lambda_X \setminus \Lambda_D$ and let n_V denote $\#\Lambda_V$. We prepare stiffness matrices $\tilde{A}$ and $\tilde{B}$ defined by

$$\begin{aligned} [\tilde{A}]_{\tilde{\alpha}\,\tilde{\beta}} &:= a(\varphi_\beta, \varphi_\alpha\,;\,\Omega) && (\alpha, \beta \in \Lambda_V)\,, \\ [\tilde{B}]_{\mu\,\tilde{\beta}} &:= b(\varphi_\beta, \psi_\mu\,;\,\Omega) && (\mu \in \Lambda_S,\ \beta \in \Lambda_V)\,. \end{aligned}$$

Here $\tilde{\alpha} \in \{1, \ldots, n_V\}$ is an index renumbered from $\alpha \in \Lambda_V$. We note that the sizes of matrices $\tilde{A}$ and $\tilde{B}$ are $n_V \times n_V$ and $n_S \times n_V$, respectively. We put $n_Z := n_V + n_S$. A stiffness matrix of the Stokes problem is given by

$$\mathcal{K} := \begin{pmatrix} \tilde{A} & \tilde{B}^T \\ \tilde{B} & -\delta D \end{pmatrix}.$$

We define vectors $\vec{b}, \vec{m}' \in \mathbb{R}^{n_Z}$ by

$$[\vec{b}]_i := \begin{cases} 0 & (1 \le i \le n_V) \\ 1 & (n_V + 1 \le i \le n_Z) \end{cases},$$

$$[\vec{m}']_i := \begin{cases} 0 & (1 \le i \le n_V) \\ [\vec{m}]_{i-n_V} & (n_V + 1 \le i \le n_Z) \end{cases}.$$

A solution space $\vec{Y} \subset \mathbb{R}^{n_Z}$ is defined by

$$\vec{Y} := \{\vec{u} \in \mathbb{R}^{n_Z}\,;\,(\vec{u},\, \vec{m}')_{\mathbb{R}^{n_Z}} = 0\}\,.$$

A space $\vec{V}_2$ in Algorithm A.1 is defined by

$$\vec{V}_2 := \{\vec{u} \in \mathbb{R}^{n_Z}\,;\,(\vec{u},\, \vec{e}^{(n_Z)})_{\mathbb{R}^{n_Z}} = 0\}\,.$$

Since the unit vector $\vec{e}^{(n_Z)}$ is not orthogonal to $\vec{b}$, Assumption A.2 is satisfied.

We obtain the following relations.

$$\begin{aligned} [\tilde{B}^T\vec{1}]_{\tilde{\beta}} &= \sum_{\mu\in\Lambda_S} b(\varphi_\beta, \psi_\mu\,;\,\Omega) = b(\varphi_\beta, 1\,;\,\Omega) \\ &= -\int_\Omega \nabla\cdot\varphi_\beta\,dx = -\int_\Gamma \varphi_\beta\cdot n\,ds \\ &= 0 \qquad (\beta \in \Lambda_V)\,. \\ [D\vec{1}]_\mu &= \sum_{\nu\in\Lambda_S} d(\psi_\nu, \psi_\mu\,;\,\Omega) = d(1, \psi_\mu\,;\,\Omega) \\ &= \sum_{K\in\mathcal{T}_h} h_K^2(\nabla 1, \nabla\psi_\mu\,;\,K) \\ &= 0 \qquad (\mu \in \Lambda_S)\,. \end{aligned}$$

We know the following fact about the kernel of the stiffness matrix $\mathcal{K}$, $N(\mathcal{K})$.

Proposition A.2 *In the Dirichlet Problem, we have*

$$\dim N(\mathcal{K}) = 1 \quad \textit{and} \quad \vec{b} \in N(\mathcal{K})\,.$$

From Proposition A.2, we can verify that Assumption A.1 is satisfied. We note that $(\vec{f_2},\, \vec{1})_{\mathbb{R}^{n_S}} = 0$ for both the GLS and penalty types. Therefore, Assumption A.3 is also satisfied. Hence, we can employ Algorithm A.1 to solve the finite element equation of the *Dirichlet Problem*.

Case of Slip boundary conditions

In the *Slip Problem*, it is required to remove rigid body rotations and pressure lifting. We only consider the homogeneous boundary data, $g_n = 0$. Let $\Lambda_W := \Lambda_X \setminus \Lambda_\Gamma$ and let n_W denote $\#\Lambda_W$. Let $\{\tilde{\varphi}_\alpha\}_{\alpha\in\Lambda_W}$ be a basis of $\vec{W}$, where $\tilde{\varphi}_\alpha(P_{\beta_0}) \cdot n_\Omega(P_{\beta_0}) = 0$ for $\beta_0 \in \Lambda_\Gamma$. This basis can be constructed as follows. Let $k(\alpha_0) \in \{1,2,3\}$ be a component index with which $\max_k |[n_\Omega(P_{\alpha_0})]_k|$ is attained for each $\alpha_0 \in \Lambda_\Gamma$. Then we define

$$\tilde{\varphi}_{[\alpha_0,\,l]} := \varphi_{[\alpha_0,\,l]} - \frac{[n_\Omega(P_{\alpha_0})]_l}{[n_\Omega(P_{\alpha_0})]_{k(\alpha_0)}}\varphi_{[\alpha_0,\,k(\alpha_0)]}$$

where $l \in \{1,2,3\} \setminus \{k(\alpha_0)\}$ for $\alpha_0 \in \Lambda_\Gamma$. On inner nodes, we define

$$[\tilde{\varphi}]_{[\alpha_0,\,l]} := [\varphi]_{[\alpha_0,\,l]} \quad (\alpha_0 \in \Lambda_G \setminus \Lambda_\Gamma,\ l \in \{1,2,3\}).$$

We recall that the rigid body rotation $v^{(i)}$ belongs to $\tilde{W}_h$, i.e., $v^{(i)}(P_{\alpha_0}) \cdot n_\Omega(P_{\alpha_0}) = 0$ for $\alpha_0 \in \Lambda_\Gamma$. Let $\{\vec{v}\,'^{(i)}\}_{i=1,2,3} \subset \mathbb{R}^{n_W}$ be vectors corresponding to the rigid body rotations and the basis $\{\tilde{\varphi}_\alpha\}$,

$$\sum_{\alpha\in\Lambda_W} [\vec{v}\,'^{(i)}]_\alpha \tilde{\varphi}_\alpha(x) = v^{(i)}(x)\,.$$

We prepare stiffness matrices $\tilde{A}$ and $\tilde{B}$ and mass matrix $\tilde{M}_1$ defined by

$$\begin{aligned}
[\tilde{A}]_{\tilde{\alpha}\,\tilde{\beta}} &:= a(\tilde{\varphi}_\beta, \tilde{\varphi}_\alpha\,;\,\Omega_h) && (\alpha,\beta \in \Lambda_W)\,,\\
[\tilde{B}]_{\mu\,\tilde{\beta}} &:= b(\tilde{\varphi}_\beta, \psi_\mu\,;\,\Omega_h) && (\mu \in \Lambda_S,\ \beta \in \Lambda_W)\,,\\
[\tilde{M}_1]_{\tilde{\alpha}\,\tilde{\beta}} &:= (\tilde{\varphi}_\beta, \tilde{\varphi}_\alpha\,;\,\Omega_h) && (\alpha,\beta \in \Lambda_W)\,.
\end{aligned}$$

Here $\tilde{\alpha} \in \{1,\ldots,n_W\}$ is an index renumbered from $\alpha \in \Lambda_W$. A stiffness matrix of the Stokes problem is also given by

$$\mathcal{K} := \begin{pmatrix} \tilde{A} & \tilde{B}^T \\ \tilde{B} & -\delta D \end{pmatrix}.$$

A solution space $\vec{Y} \subset \mathbb{R}^{n_W+n_S}$ is defined by

$$\vec{Y} := \{ \begin{pmatrix} \vec{u} \\ \vec{p} \end{pmatrix} \in \mathbb{R}^{n_W+n_S}\,;\, \vec{u} \in \mathbb{R}^{n_W}, \vec{p} \in \mathbb{R}^{n_S},$$
$$(\vec{u},\, \tilde{M}_1\vec{v}\,'^{(i)})_{\mathbb{R}^{n_W}} = 0\ (i=1,2,3),\ (\vec{p},\, \vec{m})_{\mathbb{R}^{n_s}} = 0\}\,.$$

We can see that

$$\tilde{A}\vec{v}'^{(i)} = 0 \quad (i = 1, 2, 3)\,, \tag{A.6}$$

$$\tilde{B}\vec{v}'^{(i)} = 0 \quad (i = 1, 2, 3)\,. \tag{A.7}$$

From Lemma 1.11, (i) and (A.6), we have $N(\tilde{A}) = \mathrm{span}[\vec{v}'^{(1)}, \vec{v}'^{(2)}, \vec{v}'^{(3)}]$ and $\tilde{A}$ is positive definite on $R(\tilde{A})$. From (A.7), we have

$$R(\tilde{B}^T) \subset R(\tilde{A})\,. \tag{A.8}$$

For the vector corresponding to the constant pressure, we have

$$D\vec{1} = 0\,. \tag{A.9}$$

On the other hand, $\tilde{B}^T\vec{1}$ does not vanish due to the approximation of the smooth domain Ω by a polygonal domain Ω_h. In fact,

$$[\tilde{B}^T\vec{1}]_{\tilde{\beta}} = -\int_{\Omega_h} \nabla\cdot\tilde{\varphi}_\beta\, dx = -\int_{\Gamma_h} \tilde{\varphi}_\beta \cdot n\, ds \neq 0 \tag{A.10}$$

for an index $\beta \in \Lambda_W$ corresponding to a node on Γ_h.

Proposition A.3 *In the Slip Problem, we have*

$$\dim N(\mathcal{K}) = 3\,.$$

Proof. We can rewrite the stiffness matrix $\mathcal{K}$ with a Schur complement matrix S as follows.

$$\mathcal{K} = \begin{pmatrix} I_{n_V} & 0 \\ \tilde{B}\tilde{A}^\dagger & I_{n_S} \end{pmatrix} \begin{pmatrix} \tilde{A} & 0 \\ 0 & -S \end{pmatrix} \begin{pmatrix} I_{n_V} & \tilde{A}^\dagger\tilde{B}^T \\ 0 & I_{n_S} \end{pmatrix}\,,$$

where S is defined by

$$S := \delta\, D + \tilde{B}\tilde{A}^\dagger\tilde{B}^T\,.$$

From positivity of $\tilde{A}$ on $R(\tilde{A})$, (A.8), and (A.10), we have

$$(\tilde{B}\tilde{A}^\dagger\tilde{B}^T\vec{1},\, \vec{1})_{\mathbb{R}^{n_S}} = (\tilde{A}^\dagger\tilde{B}^T\vec{1},\, \tilde{B}^T\vec{1})_{\mathbb{R}^{n_S}} \geq c||\tilde{B}^T\vec{1}||^2_{\mathbb{R}^{n_S}} > 0\,,$$

where c is a positive constant. Since D is positive definite on $\mathrm{span}[\,\vec{1}\,]^\perp$ and $\tilde{B}\tilde{A}^\dagger\tilde{B}^T$ is semi-positive definite on $\mathbb{R}^{n_S}$, S is positive definite. Combining $\dim N(\tilde{A}) = 3$ and $\dim N(S) = 0$, we get the desired result by virtue of the Sylvester law of inertia. □

From Proposition A.3, the dimension of $R(\mathcal{K})$ does not coincide with dimension of the solution space $\vec{Y}$. Hence, Assumption A.1 is not satisfied. Consequently, we cannot apply Algorithm A.1 to the finite element equation of the *Slip Problem*.

Bibliography

[1] R. Allen, K. Kennedy. *Optimizing Compilers for Modern Architectures*, Academic Press, San Diego, 2002.

[2] O. Axelsson, V. A. Barker. *Finite Element Solution of Boundary Value Problems : Theory and Computation*, Academic Press, Orlando, 1984.

[3] R. Barett, M. Berry, T. F. Chan, J. Demmel, J. Donato, J. Dongarra, V. Eijkhout, R. Pozo, C. Romine, H. Van der Vorst. *Templates for the Solution of Linear Systems : Building Blocks for Iterative Methods*, SIAM, Philadelphia, 1994.

[4] R. E. Bank, T. F. Chan. An analysis of the composite step biconjugate gradient method, *Numer. Math.*, 66 (1993) 295–319.

[5] K. J. Bathe. *Finite Element Procedures*, Prentice-Hall, New Jersey, 1996.

[6] J. R. Baumgardner. Three-dimensional treatment of convective flow in the Earth's mantle, *J. Stat. Phys.*, 39 (1985) 501–511.

[7] D. Bercovici, G. Schubert, G.A. Glatzmaier, A. Zebib. Three-dimensional thermal convection in a spherical shell, *J. Fluid Mech.*, 206 (1989) 75–104.

[8] A. Bossavit. Symmetry, groups, and boundary value problems. A progressive introduction to noncommutative harmonic analysis of partial differential equations in domains with geometrical symmetry, *Comput. Methods Appl. Mech. Engrg.*, 56 (1986) 167–215.

[9] M. Bonnet. *Boundary Integral Equation Methods for Solids and Fluids*, John Wiley & Sons, Chichester, 1995.

[10] D. Braess. *Finite Elements : Theory, fast solvers, and applications in solid mechancis* (2nd ed), Cambridge University Press, Cambridge, 2001.

[11] S. C. Brenner, L. R. Scott. *The Mathematical Theory of Finite Element Methods*, Springer-Verlag, New York, 1994.

[12] F. Brezzi, J. J. Douglas. Stabilized mixed methods for the Stokes problem, *Numer. Math.*, 53 (1988) 225–235.

[13] F. Brezzi, M. Fortin. *Mixed and Hybrid Finite Element Methods*, Springer-Verlag, New York, 1991.

[14] A. N. Brooks, T. J. R. Hughes. Streamline upwind/Petrov–Galerkin formulations for convection dominated flows with particular emphasis on the incompressible Navier–Stokes equations, *Comput. Methods Appl. Mech. Engrg.*, 32 (1982) 199–259.

[15] P. G. Ciarlet. *The Finite Element Method for Elliptic Problems*, North-Holland, Amsterdam, 1978.

[16] J. Cahouet, J. P. Chabard. Some fast 3D finite element solvers for the generalized Stokes problem, *Internat. J. Numer. Methods Fluids*, 8 (1988) 869–895.

[17] J. J. Douglas, J. Wang. An absolutely stabilized finite element method for the Stokes problem, *Math. Comp.*, 52 (1989) 495–508.

[18] G. Duvaut, J. L. Lions. *Les Inéquations en Mécanique et en Physique*, Dunod, Prais, 1972 (English translation : *Inequalities in Mechanics and Physics*, Springer-Verlag, Berlin, 1976).

[19] S. C. Eisenstat, H. C. Elman, M. H. Schultz. Variational iterative methods for nonsymmetric systems of linear equations, *SIAM. J. Numer. Anal.*, 20 (1983) 345–357.

[20] R. Fletcher. Conjugate gradient methods for indefinite systems, In *Numerical Analysis Dundee 1975*, G. A. Watson (ed.), *Letcutre Note in Mathemactics* 506, Springer-Verlag, Berlin, 1976, 73–89.

[21] L. P. Franca, S. L. Frey, T. J. R. Hughes. Stabilized finite element methods : I. Application to the advective-diffusive model, *Comput. Methods Appl. Mech. Engrg.*, 95 (1992) 253–276.

[22] L. P. Franca, R. Stenberg. Error analysis of some Galerkin least squares methods for the elasticity equations, *SIAM J. Numer. Anal.*, 28 (1991) 1680–1697.

[23] V. Girault, P. A. Raviart. *Finite Element Methods for Navier–Stokes Equations, Theory and Algorithms*, Springer-Verlag, Berlin, 1986.

[24] G. H. Golub, C. F. Van Loan. *Matrix Computations* (3rd edn), The Johns Hopkins University Press, Baltimore, 1996.

[25] P. M. Gresho, S. T. Chan, R. L. Lee, C. D. Upson. A modified finite element method for solving the time-dependent, incompressible Navier–Stokes equations. Part 1 : Theory, *Int. J. Num. Meth. Fluids*, 4 (1984) 557–598.

[26] M. R. Hestenes, E. Stiefel. Methods of conjugate gradients for solving linear systems, *J. Res. Nat. Bur. Standards*, 49 (1952) 409–436.

[27] T. J. R. Hughes, L. P. Franca. A new finite element formulation for computational fluid dynamics : VII. The Stokes problem with various well-posed boundary conditions : Symmetric formulations that converge for all velocity/pressure spaces, *Comput. Methods Appl. Mech. Engrg.*, 65 (1987) 85–96.

[28] T. J. R. Hughes, L. P. Franca, G. M. Hulbert. A new finite element formulation for computational fluid dynamics : VIII. The Galerkin/least-squares method for advective-diffusive equations, *Comput. Methods Appl. Mech. Engrg.*, 73 (1989) 173–189.

[29] Y. Iwase. Three-dimensional infinite Prandtl number convection in a spherical shell with temperature-dependent viscosity, *J. Geomag. Geoelectr.*, 48 (1996) 1499–1514.

[30] B. W. Kerninghan, D. M. Ritchie. *The C programming Language* (2nd edn), Prentice-Hall, Englewood Cliffs, 1988.

[31] W. Joubert. Lanczos methods for the solution of nonsymmetric systems of linear equations, *SIAM J. Matrix Anal. Appl.*, 13 (1992) 926–943.

[32] E. F. Kaasschieter. Preconditioned conjugate gradients for solving singular systems, *J. Comput. Appl. Math.*, 24 (1988) 265–275.

[33] *OpenMP C and C++ Application Program Interface Version 1.0 – Octorber 1998*, `http://www.openmp.org`.

[34] C. C. Paige, M. A. Saunders. Solution of sparse indefinite systems of linear equations, *SIAM J. Numer. Anal.*, 12 (1975) 617–629.

[35] C. C. Paige, B. N. Parlett, H. A. van der Vorst. Approximate solutions and eigenvalue bounds from Krylov subspaces, *Numer. Linear Algebra Appl.*, 2 (1995) 115–133.

[36] A. Quarteroni, A. Valli. *Numerical Approximation of Partial Differential Equations*, Springer-Verlag, Berlin, 1994.

[37] J. T. Ratcliff, G. Schubert, A. Zebib. Three-dimensional variable viscosity convection of an infinite Prandtl number Boussinesq fluid in a spherical shell, *Geophys. Res. Lett.*, 22 (1995) 2227–2230.

[38] J. T. Ratcliff, G. Schubert, A. Zebib. Steady tetrahedral and cubic patterns of spherical shell convection with temperature-dependent viscosity, *J. Geophys. Res.*, 101 (1996) 25473–25484.

[39] Y. Saad. The Lanczos biorthogonalization algorithm and other oblique projection methods for solving large unsymmetric systems, *SIAM J. Numer. Anal.*, 19 (1982) 485–506.

[40] Y. Saad, M. H. Schultz. GMRES : A generalized minimal residual algorithm for solving nonsymmetric linear systems, *SIAM J. Sci. Statist. Comput.*, 7 (1986) 856–869.

[41] A. Suzuki. Implementation of domain decomposition methods on parallel computer ADENART, In *Parallel Computational Fluid Dynamics : New Algorithms and Applications*, N. Satofuka, J. Periaux, A. Ecer (eds.), Elsevier-Science, Amsterdam, 1995, 231–238.

[42] A. Suzuki. Implementation of non-overlapping domain decomposition methods on parallel computer ADENA, In *Domain Decomposition Methods in Sciences and Engineering*, R. Glowinski, J. Périaux, Z-C. Shi, O. Widlund (eds.), John Wiley & Sons, Chichester, 1997, 275–282.

[43] A. Suzuki, M. Tabata, S. Honda. Numerical solution of an unsteady Earth's mantle convection problem by a finite element method, *Theoretical and Applied Mechanics*, 48 (1999) 371–378.

[44] M. Tabata. Slip boundary conditions and rigid body movements in finite element analysis, In *Advances in Numerical Mathematics; Proceedings of the Fourth Japan-China Joint Seminar on Numerical Mathematics*, H. Kawarada, M. Nakamura, Z. Shi (eds.), Gakkotosho, Tokyo, 1999, 117–124

[45] M. Tabata. Uniform solvability of finite element solutions in approximate domains, *Japan J. Indust. Appl. Math.*, 18 (2001) 567–585.

[46] M. Tabata, A. Suzuki. A stabilized finite element method for the Rayleigh-Bénard equations with infinite Prandtl number in a spherical shell, *Comput. Methods Appl. Mech. Engrg.*, 190 (2000) 387–402.

[47] M. Tabata, A. Suzuki. Mathematical modeling and numerical simulation of Earth's mantle convection, In *Mathematical Modeling and Numerical Simulation in Continuum Mechanics*, I. Babuška, P. G. Ciarlet, T. Miyoshi (eds.), *Lecture Notes in Computational Science and Engineering*, Vol. 19, Springer-Verlag, Berlin, (2002) 219–231.

[48] R. Verfürth. Finite element approximation of incompressible Navier-Stokes equations with slip boundary condition, *Numer. Math.*, 50 (1987) 697–721.

[49] S.-L. Zhang, Y. Oyanagi, M. Sugihara. Necessary and sufficient conditions for the convergence of Orthmin(k) on singular and inconsistent linear systems, *Numer. Math.*, 87 (2000) 391–405.

[50] A. Zebib. Linear and weakly nonlinear variable viscosity convection in spherical shells, *Theoret. Comput. Fluid Dynamics*, 4 (1993) 241–253.

[51] O. C. Zienkiewicz, F. C. Scott. On the principle of repeatability and its application in analysis of turbine and pump impellers, *Internat. J. Numer. Methods Engrg*, 4 (1972) 445–452.

[52] O. C. Zienkiewicz, R. L. Taylor. *The Finite Element Method* (5th edn), Butterworth-Heinemann, Oxford, 2000.